# DYNAMIC NETWORK FORMATION USING ANT COLONY OPTIMIZATION

DISSERTATION

Steven C. Oimoen, Major, USAF

AFIT/DCS/ENG/09-06

**DEPARTMENT OF THE AIR FORCE**
**AIR UNIVERSITY**

**AIR FORCE INSTITUTE OF TECHNOLOGY**

**Wright-Patterson Air Force Base, Ohio**

AFIT/DCS/ENG/09-06

**DYNAMIC NETWORK FORMATION USING ANT COLONY OPTIMIZATION**

DISSERTATION

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

Steven C. Oimoen, BA, BS, MS

Major, USAF

March 2009

AFIT/DCS/ENG/09-06

**DYNAMIC NETWORK FORMATION USING ANT COLONY OPTIMIZATION**

Steven C. Oimoen, BA, BS, MS
Major, USAF

Approved:

| | |
|---|---|
| Dr. Gilbert L. Peterson (Chairman) | 4 MAR 09<br>Date |
| Dr. Kenneth M. Hopkinson (Member) | 4 Mar 2009<br>Date |
| Robert E. Neher Jr., Lt Col, USAF (Member) | 4 Mar 2009<br>Date |
| Scott R. Graham, Major, USAF (Member) | 4 Mar 2009<br>Date |

Accepted:

| | |
|---|---|
| MU Thomas<br>M. U. Thomas<br>Dean, Graduate School of<br>Engineering and Management | 19 Mar 2009<br>Date |

AFIT/DCS/ENG/09-06

## Abstract

Military and industry are moving toward every device being network enabled and connected for reliable availability of communication and information. To make this type of system a reality, the devices must be capable of forming a network topology on their own in a dynamic environment to ensure that the correct information reaches a desired location and on-time. This research presents three contributions for solving highly dynamic (i.e. drastic change within the network) Multi-commodity Capacitated Network Design Problems (MCNDPs) resulting in a distributed multi-agent network design algorithm. The first contribution incorporates an Ant Colony Optimization (ACO) algorithm Ant Colony System (ACS) to solve the static MCNDP with weak constraints. Second, a new algorithm is developed and has the capability to dynamically adjust its exploration parameter of the solution space. This enhanced algorithm converges quickly and automatically adjusts to the dynamically changing network environment. Third, a distributed approach is created replacing the previous centralized solver. The distributed algorithm produces comparable results, but more importantly calculates the network topology in less than 20 percent of the computation time.

## Acknowledgments

I express my sincere gratitude and appreciation to my research and academic advisor, Dr. Gilbert L. Peterson. I also thank my committee members Dr. Kenneth M. Hopkinson, Lt Col Robert E. Neher, Jr., and Major Scott R. Graham for their input and advice. I thank Major June Rodriguez and Ben Rodriguez for all of their help and friendship, Gabe Greve and James Haught for their coding expertise, and my father-in-law and sister for their encouragement. Finally, special thanks go to my sons and especially my wife for their patience, understanding and support during the dissertation process and my time at the Air Force Institute of Technology.

**Table of Contents**

Page

## List of Figures

## List of Tables

Page

# DYNAMIC NETWORK FORMATION USING ANT COLONY OPTIMIZATION

## I. Introduction

Ad hoc networks are self-organizing multi-hop wireless networks that do not rely on fixed infrastructure or predetermined connectivity (Joshi, Mishra, Batta, & Nagi, 2004). This property enables ad hoc networks to be quickly deployed without advanced knowledge of the coverage area and is suited for scenarios with limited fixed infrastructure or high risk such as military communications, disaster management and law enforcement. An ad hoc network comprises a collection of geographically distributed nodes that communicate with each other over a wireless medium. Military applications provided the initial domain for the development of ad hoc networks (Rajaraman, 2002). A critical issue associated with ad hoc networks and with computer networks in general is the problem of topology control, which focuses on building and maintaining a connected topology amongst the network nodes.

The aerospace operational environment is both lethal and highly information technology dependent (Gardiner, 2006). This dependence on information technology is especially visible with respect to networks of all types and sizes. Leadership has identified that achieving and maintaining a dominant role in Cyberspace is essential (Lopez, 2006). It is critical for the Air Force to continue research in developing the means to achieve this goal. One focus has been on wireless communications technology and wireless networks (Garner, 2007). The reliability and availability demands of a mobile military network are great. In order to satisfy the demands, an infrastructure must exist to support net-centric warfare. An example is a hybrid communication network

consisting of directional links (free space optical) with directional broadcast radio frequency (RF) links. These networks provide reliable levels of network availability at a relatively low cost to several users (Erwin, 2006).

This research focuses on solving the Multi-commodity Capacitated Network Design Problem (MCNDP) using the Ant Colony System (ACS) to construct the network topology. In the following section, a background of the Network Design Problem (NDP), the (MCNDP), and Ant Colony Optimization (ACO) is given. Following this, a section devoted to the problem statement for dynamically constructing a network topology is outlined. The methodology section gives an overview of the recommended approach to solve the MCNDP under both weak and more importantly strong constraints (dynamic MCNDP). In addition, a proposed distributed solution for the MCNDP is presented in order to effectively reduce computation time developing and constructing the network topology. The last section concludes with the summary of the topics discussed within this chapter.

## 1.1 Background

In this section the NDP is defined. Next, a description of the MCNDP, a variation of the NDP, is given as this forms the problem domain and motivation of this research. In addition, the ACO algorithm is presented which is the algorithm incorporated as the solution strategy to the MCNDP.

### 1.1.1 The Network Design Problem (NDP)

The problem of efficiently and effectively transmitting data throughout a communications network must include the overall design of the subnetworks. Communication systems send information from a given source to a specified destination. The source and destination points of the system are typically referred to as nodes. A network is a communication system comprised of nodes. Nodes within the network are connected using communication links (edges). The set of current active links form the network topology. The NDP focuses on designing a flexible network while trying to achieve optimal flow or routing (Ahuja, Magnanti, & Orlin, 1993). Given a set of $N$ nodes ($x_i \in N$), each with a finite set of communication capabilities, i.e. the type of connection, of a total set of capabilities $C$. If a link (or arc) is used, then an associated fixed cost of the edge is incurred. In addition, there is an additional cost for the actual use of the arc depending on the flow. Each node may communicate with one other node that has the same capability as long as that capability is not being used by another connection. In other words, given a node $x_i$ with capabilities **A** and **B**, and a node $x_j$ with capabilities **A** and **C**, they may create a communication link of type **A** as long as neither $x_i$ nor $x_j$ is using the link **A** to connect to a different node. The solution then is to create a network in which all of the nodes $N$ are able to communicate while finding the network design that minimizes the total system cost (the sum of the design cost and the routing cost).

### 1.1.2 Multi-commodity Capacitated Network Design Problem (MCNDP)

The MCNDP is a network design problem that includes bandwidth constrained edges, multiple commodity types, and focuses on networks built of directed links, which are often dynamic (Alvarez, Gonzalez-Velarde, & De-Alba, 2005). The solution to the MCNDP is a feasible network topology that minimizes the total cost of the network (fixed edge cost, variable commodity flow cost and any penalties associated with dropped commodities) while maximizing throughput (commodities flowing throughout the network).

### 1.1.2 Ant Colony Optimization (ACO)

One of the most critical steps in determining a solution to any given problem is selecting the appropriate search algorithm. Consideration must be given to time, memory requirements, and limitations of any proposed algorithm. A stochastic search algorithm is a directed, random search technique which is useful when trying to escape being trapped in a local optimal solution which in turn leads to a sub-optimal solution. One primary issue with stochastic searches is they are not necessarily complete.

ACO is a meta-heuristic technique that has been shown to be quite successful in solving many combinatorial optimization problems (Dorigo & Stutzle, 2004). The strength of ACO is found in the foraging behavior of real ants. This behavior enables the ants to find the shortest paths between food sources and their respective nests. Ants have a tendency to deposit a substance called pheromone while traveling from their nests to food sources and via the return trip from the food source to their nests. Paths that are

marked with greater concentration of pheromone are selected with a higher probability than those paths which contain smaller amounts of pheromone. Figure 1 illustrates the ant process of determining the shortest path.

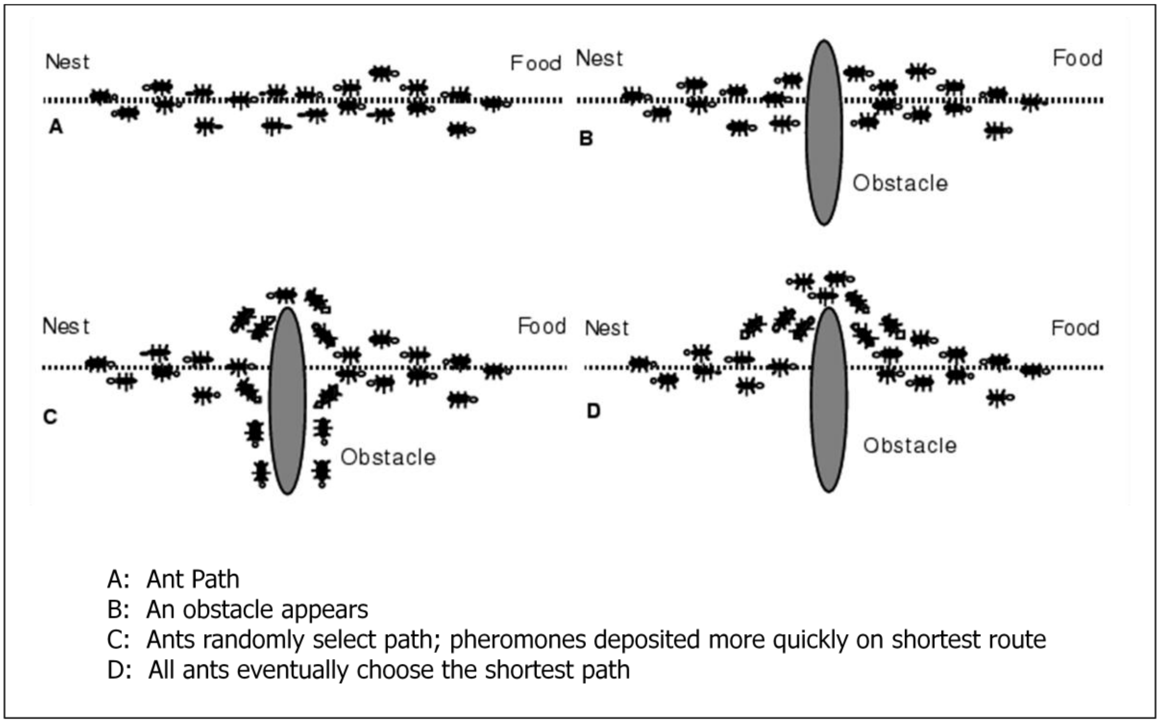


**Figure 1. Ants Finding the Shortest Path**

## 1.2 Problem Statement

The problem of dynamically linking computers in a network consisting of high-bandwidth directional links and determining the optimal routing of information is an NP-hard problem (Hochbaum, 1997). One approach to solving the dynamic network design problem, under weak dynamic constraints, is using the Mixed Integer Linear Program (MILP) approach and any linear solver (Erwin, 2006). Weak dynamic constraints include

environment changes that have minimal impact on the overall performance of the network environment. This is compared with strong dynamic constraints which have the potential to seriously impact the network performance and seriously degrade the network operating environment. Under strong dynamic constraints, solving the dynamic network design problem requires an approach that tracks changes in the network, adapts, and still computes an approximately optimal network topology in an efficient and responsive manner.

These problems exist as components of research in the field of hybrid communications known as the Topology Control Problem (also referred to as the Network Design Problem). This class of problems focuses on the construction of an optimal network for transmitting commodities between multiple nodes, often under multiple constraints such as bandwidth, directed channels, etc. The specific topology control problem focused on in this work is the MCNDP.

Initial research solved the MCNDP using ACO algorithms to construct the network topology (Oimoen, Peterson, & Hopkinson, 2008). This approach is then incorporated into a dynamic environment with both weak and strong constraints. Lastly, the approach is transformed from a central, global solver into a distributed solver where each network node independently assigned their network links.

### 1.3 Methodology

Previous research into determining efficient and effective network topologies for wireless networks focused on network flow and routing problems. Integer linear programming methods, which provide exact solutions for relatively small problem sizes,

have been used to solve directional topology control problems. Also, heuristics have been developed that provide suboptimal, but timely solutions (Garner, 2007). These methods only work when there are limited changes to the network. The approach recommended to solve the dynamic MCNDP is to use ACO to learn the network structure while handling weak to strong dynamic network alterations. ACO is a metaheuristic technique that has proven to be quite successful in solving many combinatorial optimization problems (Osman & Kelly, 1996). The foundation for this algorithm is based on the foraging behavior of real ants. Ants seek out the shortest path between a food source and their nest. Ants deposit a substance called pheromone along these paths. Paths with a stronger concentration of pheromone are chosen with a greater probability than those with lesser amounts of pheromone. In 1991, Dorigo developed the first ACO algorithm called Ant Cycle, Ant Quantity, and Ant Density (Dorigo, Maniezzo, & Colorni, 1991).

A total of four different ACO methodologies are discussed titled Ant Colony System Standard (ACSS), Ant Colony System Estimation (ACSE), Distributed ACSS (DACSS), and Distributed ACSE (DACSE). In addition, each of these four methodologies is further broken down into a dynamic category (1, 2, and 3). All of the ACO approaches use a pseudo-random proportional rule to balance exploration and exploitation of the solution space and perform appropriately with respect to weak and strong dynamic change.

### 1.3.1 Ant Colony System Standard (ACSS)

The first is the base ACO algorithm, ACSS, which builds the network structure and uses network flow routing strategies (Garner, 2007) to solve the routing problem.

Each ant in the colony constructs a network topology across which all commodities are routed. From this network and routing, the complete cost of the network (fixed edge cost, commodity flow cost, and any penalties associated with dropped commodities) is evaluated.

### 1.3.2 Ant Colony System Estimated (ACSE)

The second approach incorporates multiple heuristics within the ACO implementation. These heuristics approximate the cost to actually route the commodities throughout the network. The heuristic include the fixed edge cost of an edge, the variable edge cost of the edge, the value of a commodity, and the capacity of the edge. The benefit of this approach is eliminating the need for each ant solution to route the commodities in order to evaluate the proposed network topology. This approach significantly reduced the overall run time of the algorithm to solve the MCNDP.

### 1.3.3 ACSS/ACSE - Dynamic Environment

It is absolutely essential that a topology control algorithm adapt to a changing network environment. This area of the research focuses on implementing ACSS and ACSE methodologies within the same MCNDP domain, but in a strongly dynamic environment. Again, three dynamic categories were created. Category 1 (Baseline) consists of the two original algorithms ACSS and ACSE. No changes are made to the algorithms to handle the strong dynamics; however, each is monitored to evaluate how the Ant Colony System (ACS) algorithm in general handles network change. Category 2 (Restart) algorithms are modified in order to re-initialize the pheromone matrix whenever

a change to the network occurs and provides a control algorithm group for comparisons. Category 3 (Dynamic) algorithms include a dynamic feature which monitors for weak and strong dynamic change and automatically adapts to modifications introduced throughout the network. All approaches are monitored as they responded to changes occurring in the network. The network changes include edge availability, fixed edge costs, variable commodity flow costs, commodity demand, and edge capacity (bandwidth).

### 1.3.4 Distributed ACSS/ACSE

In addition to adapting to strong dynamic change within a network, a topology control algorithm must be able to respond in an efficient and timely manner. This research effort modifies the previous implementation which uses a centralized solver to construct network topologies. The new approach solves the network topology problem in a distributed manner. The new distributed approach constructs the network topology using a process where each node acts independently of each other. Each node runs its own ACSS/ACSE algorithm. Each node now makes its own decision as to which edges are being selected to connect to other nodes within the network. Once a node has completed its selection process, the selected outgoing edges are provided to a higher level abstraction which essential serves as a network event manager. The network event manager consolidates each node's outgoing edge list into a master outgoing edge list which forms the overall network topology. The network topology (i.e. network graph) is then passed to the selected routing strategy for a complete network evaluation. The edge selection portion of the solution to the network topology is generated at the node level

and then consolidated and evaluated at a global level to properly evaluate the network topology constructed.

### 1.3.5 Results

During the first phase of the research, both the ACSS and ACSE approaches were compared with previous solutions for solving the MNCDP. The first comparison was with a MILP approach to solve the MCNDP (Erwin, 2006). The second comparison was with a methodology that utilized flow networks and maximum flow networks for solving the MCNDP (Garner, 2007). The ACSS and ACSE algorithms produce solutions with comparable and at times improved solutions with respect to network topology costs.

During the second phase, ACSS and ACSE were integrated into a dynamic environment. Three categories of each algorithm are compared. Most noteworthy, the dynamic algorithm is able to respond to changes within the network and converge quickly on a solution for the new network.

Phase 3 generated the distributed implementations of both ACSS and ACSE. Most importantly, the distributed solvers (DACSS and DACSE) produce very similar results to the ACSS and ACSE approaches, but drastically reduce the computation time to generate a MCNDP solution.

## 1.4 Summary

This chapter defined the Network NDP, MCNDP, and ACO. Then, the problem statement is presented and the associated challenges to be met. The methodology for this research is introduced in Section 1.3 which includes two ACO extensions ACSS and

ACSE, the dynamic MCNDP environment (to include the three dynamic categories), and the two distributed solvers (DACSS and DACSE).

Chapter 2 provides the necessary background and literature review in solving the complex problem of network topology construction for the MCNDP. Chapter 3 describes the methodology in detail for ACSS, ACSE, the dynamic MCNDP environment, and the distributed solvers (DACSS and DACSE). Chapter 4 presents the results of using ACSS and ACSE to solve the MCNDP in a static network and dynamic network environment as well as the results of the distributed approaches, DACSS and DACSE. Chapter 5 provides an overall summary of the research completed and recommendations for future research efforts.

# II. Literature Review

This research focuses on the design of robust and reliable communications networks, specifically for solving the Multi-commodity Capacitated Network Design Problem (MCNDP). The literature review introduces relevant research and literature that currently exists with primary focus on network topology control and the MCNDP. First, the Network Design Problem (NDP) is discussed, followed by a variation called the MCNDP. Then, previous research efforts used to solve the MCNDP are examined, including Mixed Integer Linear Program (MILP), Maximum Flow Methods, Genetic Algorithms (GAs), and Tabu Search Algorithms. Then, Ant Colony Optimization (ACO) is presented as this research incorporates this particular search algorithm. Next, Bayesian Networks and the learning of them using ACO are presented as this methodology forms the basis for using ACO in solving the MCNDP. Finally, applications with respect to dynamic environments and distributed processing are provided as motivation for the research into the distributed MCNDP solvers.

## 2.1 Background

The Topology Control Problem defines a problem and solution space in how to connect nodes within a network to maximize overall network communication performance. The topology selection, considering the total number of possibilities, greatly impacts the network's performance. Therefore, network topology control is a primary concern to be addressed.

Topology control is an autonomous reconfiguration process (Llorca, 2005). Throughout the lifecycle of the network, the process is continually repeated and consists of these five states (Garner, 2007):

- Link state examination
- Collection of link state information
- Solution computation
- Solution distribution
- Reconfiguration

A computer network contains a set of nodes and a set of links. Computer network nodes include devices such as servers, routers, switches, and workstations; whereas fiber optic wires, the radio frequency (RF) medium, and free space optical links are a few examples of links (also known as edges or arcs). Given a set of nodes and edges, and an associated list of network traffic requirements, the objective is to find the least expensive physical network connectivity with a corresponding capability to meet all network traffic requirements subject to a predefined set of criteria. Simply stated, minimum cost with maximum bandwidth. Therefore the primary goal to obtain a feasible network topology is constrained by the number of network nodes, the available set of links (edges) and their associated properties, and the traffic requirement demands (also identified as commodities). The optimization seeks to balance these constraints.

## 2.2 Network Design Problem (NDP)

Solving the Network Design Problem (NDP) has been approached using several different methodologies. One of the most basic communications network models is the minimum cost flow problem which computes the minimum cost flow between two nodes based on the edge cost. The objective is to connect the nodes using the edges which produce the shortest path for the network. The shortest path problem is similar to the minimum cost flow problem but incorporates an upper and lower bound on each arc's flow. Again, the algorithm finds the shortest path between two nodes. For a centralized multipoint network (i.e. a tree network), the network design problem is viewed as a common combinatorial optimization problem called the constrained minimal spanning tree and is NP-complete (Lo & Chang, 2000). Based on the problem requirements, the network design problem is modeled as a Degree Constrained Minimum Spanning Tree (DCMST), a probabilistic undirected graph, and a mixed integer linear program to identify just a few. The DCMST approach to solving the NDP consists of nodes, arcs, and degree constraints for each node in the network. The degree constraint is imposed to limit the total number of arcs that are connected to each node. The objective function only uses the fixed cost of the network in its calculation and commodities are not included in the problem formulation. Using a probabilistic undirected graph, the network model is based on a graph that contains nodes, arcs, and link reliability. The capacity of the links is not included and only the fixed costs are considered. Again, commodities are not included in the model and therefore variable costs (costs incurred to route commodities across the

arcs) are not part of the objective function. Also, the uncapacitated NDP allows all arcs to have unlimited capacity, allowing for all commodities to flow across the network.

### 2.3 Multi-commodity Capacitated Network Design Problem (MCNDP)

A variant of the traditional NDP is the Multi-commodity Capacitated Network Design Problem (MCNDP). Network design problems have wide-spread applications. Specifically, the MCNDP is found in communications and transportation network planning (Gendron, Crainic, & Frangioni, 1998). In most situations, a requirement exists to send flows to satisfy demands using edges with known capacities. There is a fixed cost to use an edge and an additional, variable flow cost for the actual commodity using a particular edge. Solving the MCNDP is a much greater challenge than solving the uncapacitated problem.

The MCNDP is an NP-complete problem (Alvarez, Gonzalez-Velarde, & De-Alba, 2005) which provides a model to capture many key components of a real-world communications network. The MCNDP is similar to an uncapacitated NDP with capacity limits imposed upon each arc.

The formulation of an uncapacitated NDP to include capacity constraints is represented as (Erwin, 2006):

Let $N$ denote the set of nodes, $K$ the number of commodities, and $F$ the number of interface types.

Let $(i, j, f)$ denote the arc connecting node $i$ to node $j$ by the interface type $f$.

Let $A$ denote the node-incidence matrix where $a_{if} = 1$ if node $i$ is incident to node $j$ via interface type $f$, and $a_{if} = 0$ otherwise.

Let $x_{ijf}^{k}$ denote the fraction of the required flow of commodity k to be routed from the source $s^k$ to the destination $d^k$ that flows on arc $(i, j, f)$.

Let $y_{ijf}$ denote the binary variable indicating whether arc $(i, j, f)$ is selected as part of the network topology

Let $v_{ijf}^{k}$ denote the per unit cost of commodity k on arc $(i, j, f)$ multiplied by the flow requirement for that commodity.

Let $c_{ijf}$ denote the fixed cost of including arc $(i, j, f)$, in the network.

Let $u_{if}$ denote the number of interface of type $f$ at node $i$.

Let $b^k$ denote the required bandwidth for commodity k.

Let $cap_{ijf}$ denote the capacity of arc $(i, j, f)$.

$$\text{Minimize} \quad \sum_{\{k,(i,j,f):\, a_{ijf}=1\}} v_{ijf}^{k} x_{ijf}^{k} + \sum_{\{(i,j,f):\, a_{ijf}=1\}} c_{ijf}\, y_{ijf} \qquad [1]$$

subject to

$$\sum_{\{j,f:\, a_{ijf}=1\}} x_{ijf}^{k} - \sum_{\{j,f:\, a_{jif}=1\}} x_{jif}^{k} = \begin{cases} 1 & \text{if } i = s^{k} \\ -1 & \text{if } i = d^{k} \\ 0 & \text{otherwise} \end{cases} \qquad \forall\ i \in N, k = 1,\ldots,K \qquad [2]$$

$$\sum_{k} r^{k} x_{ijf}^{k} \le cap_{ijf} \qquad \forall (i,j,f) \in A \ \ni a_{ijf} = 1 \qquad [3]$$

$$\sum_{j \in N} y_{ijf} \le u_{if} \qquad \forall i \in N, f = 1,\ldots,F \qquad [4]$$

$$x_{ijf}^{k} \le y_{ijf} \qquad \forall (i,j,f) \in A \ \ni a_{ijf} = 1, k = 1,\ldots,K \qquad [5]$$

$$y_{ijf} = y_{jif} \qquad \forall (i,j,f) \in A \ \ni a_{ijf} = 1 \qquad [6]$$

$$x_{ijf}^{k} \ge 0 \qquad \forall (i,j,f) \in A \ \ni a_{ijf} = 1,\ k = 1,\ldots,K \qquad [7]$$

$$y_{ijf} \text{ is binary} \qquad \forall (i,j,f) \in A \ \ni a_{ijf} = 1 \qquad [8]$$

The formulation is designed to ensure the network has all nodes capable of communicating while ensuring the total system cost (to include design cost and routing cost) is minimized, subject to all constraints identified. The objective function (1) identifies exactly what is being optimized. Again, the specific objective is to minimize the total cost of the network to include the fixed cost to use a link within the network and the commodity flow cost, the cost to actually route a commodity across a specific link. Equation (2) implies $x_{ijf}^{k} \le 1$. Equation (3) represents the capacity constraints of an arc whereas equation (4) identifies the interface degree constraints. An assumption of this

model is that if node *i* is connected to node *j*, then network traffic flows in each direction (bidirectional), that is, from *i* to *j* or from *j* to *i*, requiring equation (6).

The formulation of the MCNDP has a potential shortfall. A feasible solution only exists if a topology with sufficient link capacity to satisfy all commodity bandwidth requirements is available. If that condition is not met, then there is no feasible topology, and the model does not produce a solution. In such a case, the model selectively drops the commodity constraints until a feasible solution is identified. Specifically, certain commodities must be excluded to allow sufficient capacity in order to satisfy the bandwidth requirements of the remaining commodities. In previous studies (Erwin, 2006), the assumption made is that there is no value in satisfying any less than 100% of a commodity's demand. Therefore an entire commodity is dropped rather than allowing partial satisfaction of the commodity's demand. When a commodity is omitted, an undesirable effect is the failure to send information from one user to another, Therefore, when it becomes necessary to drop commodities; it is done in increasing order of priority--dropping the lowest priority commodity first. A commodity's priority is considered to be directly proportional to its bandwidth requirement. Therefore a higher bandwidth requirement equates to a higher priority level. An additional binary variable $m^k$ is introduced which denotes the decision to omit commodity $k$ from consideration. If $m^k$ = 1, then commodity $k$ is dropped. When a commodity is dropped, a very large penalty is included in the objective function associated with dropping commodity $k$ so that commodities are dropped only to achieve feasibility. The revised formulation is as follows:

$$\text{Minimize} \quad \sum_{\{k,(i,j,f):\, a_{ijf}=1\}} v_{ijf}^k x_{ijf}^k + \sum_{\{(i,j,f):\, a_{ijf}=1\}} c_{ijf}\, y_{ijf} + \sum_k 1000 r^k m^k \qquad [9]$$

Equation (2) is replaced by Equation (10), and Equation (11) is added. Equation (10) allows the omission of commodities.

$$\sum_{\{j,f:\, a_{ijf}=1\}} x_{ijf}^k - \sum_{\{j,f:\, a_{jif}=1\}} x_{jif}^k = \begin{cases} 1-m^k & \text{if } i = s^k \\ -1+m^k & \text{if } i = d^k \\ 0 & \text{otherwise} \end{cases} \qquad \forall\ i \in N, k = 1,\ldots,K \qquad [10]$$

$$m^k \text{ is binary} \qquad \forall k = 1,\ldots,K \qquad [11]$$

## 2.4 MCNDP Solution Methods

Several methodologies have been researched to solve the MCNDP. Three specific areas of research have focused on simplex-based cutting plane methods, Lagrangian relaxation and heuristics (Gendron, Crainic, & Frangioni, 1998). In addition, the MILP approach (Erwin, 2006), a model based on flow networks and maximum flow algorithms (Garner, 2007), a multiobjective evolutionary algorithm (Kleeman, Lamont, & Graham, 2007) and a tabu-search based algorithm (Zaleta & Alvarez-Socarras, 2004) have all been applied to solve the MCNDP. A brief discussion of each approach is presented.

### 2.4.1 Mixed-Integer Linear Programming (MILP)

The MCNDP solution method uses a MILP to solve the NDP (Erwin, 2006). MILP techniques provide an optimal solution, but are costly with respect to time (Ahuja,

Magnanti, & Orlin, 1993). Since this method did not scale relative to network size, three potential extensions all using a Degree Constrained Minimum Spanning Tree (DCMST), which ensures connectedness, were created (Erwin, 2006). The first two approaches add edges to nodes in non-decreasing order or non-increasing order, titled heuristic 1 and heuristic 2. The third approach is titled combo because it utilized both the MILP and the DCMST. These methods did not guarantee optimality, but runtime is reduced.

Solving this MILP provides optimal solutions to the MCNDP, specifying which edges (links) are included in the network topology and the associated routing (identifying which links each commodity flow on). The MILP is solved using any linear solver application. The specific MILP tool was XPRESS-MP which employs the dual simplex, primal simplex, or the Newton Barrier method to solve the relaxed linear program (Erwin, 2006).

### 2.4.2 Maximum Flow Methods

The maximum flow approach to solving the MCNDP consists of two main phases (Garner, 2007):

- Selecting a combination of commodities
- Determining if the selected combination fits within the network

Garner’s research is based on two front-end methods, which utilize one of four maximum flow methods, providing a total of eight unique approaches to solving the topology control problem and is illustrated in Figure 2.

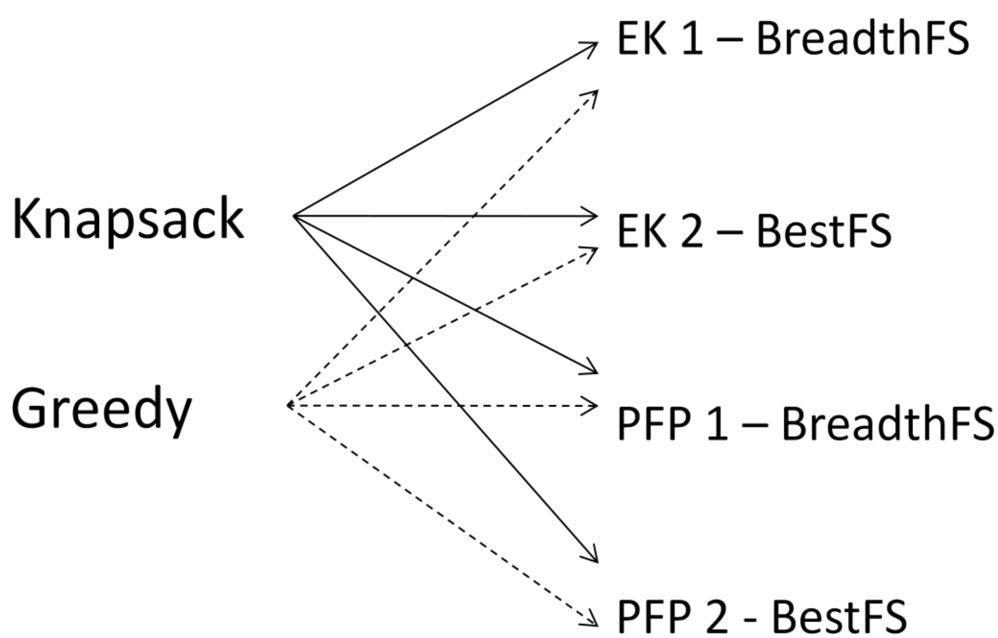


**Figure 2. Methods for Solving the NDP (5)**

Flow networks are a generalization of communications networks. A network flow is "an abstract entity that is generated at source nodes, transmitted across edges, and absorbed by sink nodes" (Kleinberg & Tardos, 2006). A flow network is defined as a graph $G = (V, E)$ where $V$ is the set of vertices (or nodes) and $E$ is the set of edges (or links, arcs). Every edge contained in the set $E$ is directional and capacitated. Also, the set $V$ contains two types of nodes, $s$ and $t$, representing the source and sink respectively. All traffic originates and exits at the source node and must eventually enter the sink node. All remaining nodes are intermediate allowing traffic to flow through them.

Garner's first approach to selecting a combination is based upon the knapsack problem, trying to maximize the value of a set of items while adhering to a strict cost (weight) constraint. The knapsack formulation is recursive, saving time by minimizes the amount of flow the net flow routine has to solve.

His second method is a greedy technique, always selecting the best option available at any given time. It starts by selecting the best commodity from the list of commodities that have yet to be explored. The list is already sorted; therefore it only has to check the next commodity in the list. Once all commodities have been explored, the solution is stored in the “solved” list (Garner, 2007).

The second phase is net flow computation. Garner narrowed his selection to the Edmonds-Karp and Pre-flow Push algorithms (Garner, 2007). Each algorithm functions with fixed edges and a set of potential edges. The critical component is the graph (i.e. the network). Each network object contains a list of nodes and a list of commodities. Each node contains a list of regular edges and potential edges. The heuristic options that Garner included are the breadth-first search (BFS) and a best-first search (BestFS).

The Edmonds-Karp Maximum Flow algorithm is a variant of the Ford-Fulkerson Maximum Flow algorithm (Ford & Fulkerson, 1962). The method finds a path from the source to the sink, pushes flow along that path, updates edge capacities and repeats until there are no additional augmenting paths available. For the second version, Garner replaces the BFS with a BESTFS to search for the augmenting path.

The Pre-flow Push uses a combination of push and pull techniques rather than augmenting flow from the source to the sink. A series of push and pull operations are performed recursively until the sink has received $c$ units of flow and the source has pushed $c$ units. This process continues until all incoming edges of the sink are saturated (Garner, 2007). The first Pre-flow Push uses BFS as its heuristic starting at the node with the highest priority and has potential edges available for consideration. The second Pre-

flow Push uses BestFS as the heuristic but determines if a demand is met without potential edges.

### 2.4.3 Genetic Algorithm

A genetic algorithm (GA) is a probabilistic search algorithm which is applied to combinatorial optimization problems. A GA simulates the real-world principles of natural selection and "survival of the fittest." The GA accomplishes these processes by taking an initial population and applying genetic operators during each reproduction.

Each individual in the population is encoded into a string called a chromosome which represents a potential solution to a given problem. The fitness of an individual chromosome is evaluated with respect to a given objective function. Individuals (or chromosomes) which are highly fit are provided greater opportunities to reproduce by swapping pieces of their genetic information with other highly fit individuals using a crossover operator. This operation produces new "offspring" solutions which share information from both contributing parents. Mutation may be applied after crossover by altering some specific genes in the string (chromosome). The offspring might replace the entire population or replace less fit individuals. This process is repeated until a satisfactory solution is found. The basic steps of a simple GA are listed below (Beasely & Chu, 1996) in Figure 3:

```
Generate an initial population;
Evaluate fitness of individuals in the population;
repeat
        Select parents from the population;
        Recombine (mate) parents to produce children;
        Evaluate fitness of the children;
        Replace some or all of the population by the children;
until a satisfactory solution has been found;
```

**Figure 3. Basic Steps of a Simple GA (20)**

A critical first step in designing a genetic algorithm is to devise a suitable representation of the data (i.e. the chromosome). Typically, a 0-1 binary representation is a good choice for a chromosome.

A multi-objective evolutionary algorithm was developed to solve a similar type of network problem (Kleeman, Lamont, & Graham, 2007). This formulation followed the MILP approach (Erwin, 2006) and had considerably slower run times due to the number of generations required (Garner, 2007). The two primary objectives of the multi-objective evolutionary approach are total cost and average number of hops. However, the mean test results perform better than the MILP approach.

A Multi-Objective Hybrid Genetic Algorithm (MOHGA) was also proposed to solve the Multi-commodity Capacitated Network Design Problem (MCNDP) (Lo & Chang, 2000). The methodology incorporates the use of a subpopulation in the selection. The MOHGA is able to find most nondominated solutions in the feasible solution space.

### 2.4.4 Tabu Search

Tabu search is traced back to 1970 and was first presented by Glover in 1986 (Aarts, E; Lenstra J, 1997). In the tabu search category of meta-heuristics, the essential idea is to restrict search moves to points already visited in the (usually discrete) search space, at least for the upcoming few steps. Tabu search is a meta-heuristic process to solve the combinatorial optimization problems (Zaleta & Alvarez-Socarras, 2004). Tabu search further improves the performance of a local search method by using memory structures. Tabu search uses a local or neighborhood search procedure to iteratively move from solution $x$ to a solution $x^*$ within the neighborhood of $x$, until some stopping criteria has been met. In order to escape from a local optimal solution, tabu search modifies the neighborhood structure of each solution as the search progresses. The solutions added to $N^*(x)$, the new neighborhood, are found using the special memory structure. The search iteratively moves from a solution $x$ to a solution $x^*$ in $N^*(x)$. The short-term memory which determines the solutions in $N^*(x)$ is the tabu list. A tabu list contains the solutions that have been visited in the recent past (less than $n$ moves ago, where $n$ is the tabu tenure). Solutions in the tabu list are not contained in $N^*(x)$.

An efficient procedure to determine solutions to the MCNDP is additionally presented (Zaleta & Alvarez-Socarras, 2004). Their design of the problem is formulated as a mixed-integer programming problem and solved using a Tabu Search meta-heuristic. Their solution methodology is based on a four-step process: obtain an initial solution, perform a local search, execute diversification moves and then repeat steps 2 and 3 a

predefined number of times. Their solutions to the problem are identified in a relatively short period of time.

For computer network problems, previous research also focused on implementing ACO approaches to assist with routing (Laxmi, Jain, & Gaur, 2006), however this work goes beyond this and uses ACO to build the network topology. The process of constructing this topology builds on the methods developed in applying ACO techniques to construct Bayesian Networks (deCampos, Fernandez-Luna, Gamez, & Puerta, 2002). By extending the Bayesian network ACO techniques and augmenting them with heuristics suitable for the MCNDP, ACO provides a significant improvement in constructing the overall network topology.

## 2.5 Ant Colony Optimization (ACO)

In 1991, Dorigo developed the first Ant Colony Optimization (ACO) algorithm called Ant System and later proposed three variations titled Ant Cycle, Ant Quantity, and Ant Density (Dorigo, Maniezzo, & Colorni, 1991). ACO is further extended with additional implementations called Elitist Ant System (Dorigo, Maniezzo, & Colorni, 1996), Rank-Based Ant System (Bullnheimer, Hartl, & Strauss, 1999), Max-Min Ant System (Stutzle & Hoos, 1997), and Ant Colony System (Dorigo & Gambardella, 1997).

The first ant colony optimization algorithm "Ant System" (AS) was developed in 1991 (Dorigo, Maniezzo, & Colorni, 1991). AS was tested against several Traveling Salesman Problems (TSPs) and was shown to be quite successful in solving problems of size less than 75 cities. This original algorithm is a set of three algorithms called Ant-Cycle, Ant-Density, and Ant-Quantity. The primary difference between these variations

is how pheromone is deposited. Ant-Cycle performs the best of the three. In all implementations, pheromone evaporated at a predetermined rate from one cycle to the next. Through evaporation, the algorithm is able to explore and prevent premature convergence to sub-optimal solutions.

The Ant System methodology evolved into the ACO metaheuristic as a common framework for ant colony algorithms. In ACO, each ant builds a possible solution to the problem by moving through a finite sequence of neighbor states (nodes). Moves are selected by applying a stochastic local search directed by the ant's internal state, problem-specific local information and shared information in the form of pheromone.

Pheromone is modeled by means of a matrix $\tau$, where $\tau_{ij}$ contains the level of pheromone deposited in the arc from node *i* to node *j*. An ant *k* at node *i* selects the next node *j* to visit with probability:

$$p_k(i,j) = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{u \in J_k(i)}[\tau_{iu}]^{\alpha}[\eta_{iu}]^{\beta}} & \text{if } j \in J_k(i) \\ 0 & \text{otherwise} \end{cases} \quad [12]$$

where $\eta_{ij}$ represents heuristic information about the problem; $J_k(i)$ is the set of neighbor nodes of node *i* that have not yet been visited by the ant *k*; $\alpha$ and $\beta$ are two parameters that determine the relative importance of the pheromone with respect to the heuristic information. For example, in the TSP, $\eta_{ij} = 1/d_{ij}$, with $d_{ij}$ being the distance between cities *i* and *j*. The identified best settings for the alpha parameter is 1 and for the beta parameter a value from 2 to 5 (Dorigo & Stutzle, 2004).

A different transition rule (for complex models) introduces another parameter, as a trade-off between exploitation and exploration. The next node *j* to visit is chosen as

$$j = \begin{cases} \arg\max_{u \in J_k(i)} \left\{ [\tau_{iu}]^{\alpha} [\eta_{iu}]^{\beta} \right\} & \text{if } q \leq q_0 \\ J & \text{otherwise} \end{cases} \qquad [13]$$

where $q$ is a random number uniformly distributed in (0.0...1.0); $q_0$ is the parameter that determines the relative importance of exploitation versus exploration ($0 \leq q_0 < 1$); $J \in J_k(i)$ is a node randomly selected according to the probabilities in equation 13 (Dorigo, Maniezzo, & Colorni, 1991).

During an iteration of the algorithm, each ant using the transition rule (Equation 15), progressively builds a solution (path). The pheromone matrix $\tau$ is updated by only allowing the ant that constructed the best solution to reinforce the level of pheromone in the arcs that are part of the best solution, $S^+$, obtained so far. This directs the search in the neighborhood of the best solution. The pheromone update rule is:

$$\tau_{ij} = (1-\rho)\,\tau_{ij} + \rho \Delta \tau_{ij}^{bs} \qquad \forall (i, j) \in T^{bs} \qquad [14]$$

where $\rho$ is the pheromone evaporation parameter that controls the decay of pheromone. Once all the ants have completed their tour, the pheromone for each edge is updated. Each pheromone value is decreased using the update rule. Over the years, many improvements have been introduced to ACO algorithms which have turned it into an effective algorithm for solving NP-hard problems. One of the greatest improvements is a feature not found within the real ants -- using information at the “global” level to bias the

search around good solutions. In 1996, Dorigo and Gambardella introduced the Ant Colony System (ACS) which focused on intensifying the solution within the neighborhood of the best solution (Dorigo & Gambardella, 1997). ACS differs from previous ACO algorithms in that ants update the pheromone trails while building solutions. The updating is accomplished using a local update rule:

$$\tau_{ij} = (1-\xi)\tau_{ij} + \xi\tau_0 \quad where\ 0 < \xi < 1 \qquad [15]$$

There have been some extensions to include Elitist Ant System (Dorigo, Maniezzo, & Colorni, 1996), Max-Min Ant System (MMAS) (Stutzle & Hoos, 1997) and Rank-Based Ant System (RBAS) (Bullnheimer, Hartl, & Strauss, 1999). For the Elitist Ant System, for every iteration, the global best solution deposits pheromone in addition to all the other ants. The MMAS incorporated maximum and minimum pheromone deposits and all edges are initialized with the maximum pheromone value. The Rank-Based Ant System ranks all solutions according to their fitness. The actual amount of pheromone deposited is applied using a weighting for each solution. This ensures more optimal solutions deposit more pheromone than less desirable solutions.

## 2.6 Bayesian Networks

Bayesian Networks (BNs), also known as Belief or Causal Networks, represent knowledge in a directed acyclic graph (DAG) to efficiently manage the dependence and independence relationships existing amongst random variables within a given problem domain (deCampos, Gamez, & Puerta, 2002). A BN representation consists of two

components a) a graphical structure (directed acyclic graph), and b) a set of parameters (collection of conditional probabilities) at each node.

A BN is a DAG $G = (V, E)$, where the set of nodes $V = \{x_1, x_2, . . ., x_n\}$ represents the system variables and $E$, a set of edges, represents the direct dependence relationships among the variables. A set of parameters is also maintained for each variable in $V$, which are the conditional probability distributions. For each variable $x_i$ in $V$ we have a family of conditional distributions $P(x_i|Pa(x_i))$, where $Pa(x_i)$ represent the set of parent nodes of the variable $x_i$. Using these conditional distributions, the joint distribution over $V$ is:

$$P\ x_1, x_2, ..., x_n\ = \prod_{i=1}^{n} P\ x_n \mid Pa\ x_i \qquad [16]$$

Learning a BN from data requires learning both the structure of the graph and the conditional probability tables at each node. The BN learning problem is: given a set of training samples, $d \in D$, where each sample $d = \{v^1,\ldots, v^n\}$ is an assignment of attributes for each node in $V$, find the BN that best matches $D$. The actual learning of the BN, in general, is an NP-hard problem and therefore, the problem is typically addressed using a heuristic method (Chickering, Geiger, & Heckerman, 1996).

The majority of BN learning algorithms apply standard search techniques such as greedy (stochastic or deterministic), hill-climbing, or others such as simulated annealing, genetic algorithms, or tabu search (deCampos, Gamez, & Puerta, 2002). More recent techniques focus on incorporating the Ant Colony Optimization algorithm as the search engine for the BN learning problem (deCampos, Fernandez-Luna, Gamez, & Puerta, 2002) (deCampos, Gamez, & Puerta, 2002). A typical methodology for solving this

problem incorporates a scoring function, *f* that evaluates each network with respect to the training data and then searches for the best network based upon this score.

A highly desirable and critical property of a metric is that the scoring function is decomposed in the following manner:

$$f(G:D) = \sum_{i=1}^{n} f(x_i, Pa(x_i) : N_{x_i, Pa(x_i)}) \qquad [17]$$

where $N_{x_i, Pa(x_i)}$ are the statistics of the variable $x_i$ and $Pa(x_i)$ in *D* (the number of instances in *D* that match each possible instantiation of $x_i$ and $Pa(x_i)$).

A respected algorithm for learning BNs is the K2 algorithm (Cooper & Herkovits, 1992). This particular algorithm utilizes a Bayesian scoring metric that measures the joint probability of a BN *G* and a training set *D*. The expression is:

$$P(G, D) = P(G) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \qquad [18]$$

where $r_i$ is the number of possible values for the variable $x_i$; $q_i$ is the number of possible instantiations for the variables in $Pa(x_i)$; $N_{ijk}$ is the number of cases in *D* where the variable $x_i$ has its $k^{th}$ value and $Pa(x_i)$ is instantiated to its $j^{th}$ value, and finally

$$N_{ij} = \sum_{k=1}^{\tau_i} N_{ijk} \qquad [19]$$

based upon a requirement that the ordering $\theta$ as an input to the K2 algorithm and given a uniform for $P(G)$, it is possible to maximize the metric (deCampos, Gamez, & Puerta,

2002). This is accomplished by evaluating each variable and parent set $Pa(x_i)$ separately. The K2 algorithm evaluates the given ordering $\theta$, and for each variable begins with an empty parent set and then adds to the set the variable that is lower ranked than $x_i$ and is computed as:

$$f_{k2}\left(x_i, Pa\left(x_i\right)\right) = \log\left(\prod_{j=1}^{q_i} \frac{\left(r_i - 1\right)!}{\left(N_{ij} + r_i - 1\right)!} \prod_{k=1}^{r_i} N_{ijk}!\right) \qquad [20]$$

The K2SN algorithm extends the original K2 algorithm by removing of the requirement for a user provided node ordering (deCampos & Peurta, 2001). The K2SN algorithm begins with an empty graph and iteratively identifies the best node to add. At each step, for every variable that is still not in the graph, the best parent set is chosen from the variables which already exist in the graph and the variable producing the best K2 score is then added to the graph in addition to the arcs defining its parent set.

### 2.6.1 Bayesian Networks Using ACO

In learning a Bayesian Network using ACO, the node ordering component of the K2SN search is guided by a colony of ants and performs in the space of orderings (permutation of the network variables). Standard ACO notation is used with $\eta_{ij}$ is the heuristic value associated with edge $(i,j)$; $\tau_{ij}$ is the amount of pheromone stored in edge $(i,j)$; $\tau_0$ is the initial amount of pheromone; and $\rho$ is the parameter that controls the pheromone evaporation (decay). The heuristic component $\eta_{ij}$ uses the K2SN heuristic. The heuristic information is dynamic since using K2SN the scoring function $f(x_j,Pa(x_j))$ associated with moving from node $i$ to node $j$ relies on the nodes previously visited.

Therefore, each ant performs a K2SN search weighted by the amount of pheromone deposited in each arc. The representation of the problem consists of a search space that is the set of permutations of the variables. A complete graph is defined over the variables, suggesting that it is always possible to reach node *j* from node *i* for every pair of nodes (*i*,*j*) (similar to the TSP, although asymmetric since the score going from *i* to *j* may not be the same as going from *j* to *i*. The heuristic is the log-likelihood log *P*(*G*,*D*), yielding a negative result and therefore $\eta_{ij} = \frac{1}{|f(x_j, Pa(x_j))|}$, with $Pa(x_j)$ being computed using the K2 metric. Pheromone is initialized using a small amount for every link in the graph. The actual amount if calculated as a function based upon the "goodness" of a solution to the problem using a greedy algorithm $(\tau_0 = \frac{1}{n \cdot |f(S_{K2SN})|})$, where *n* is the number of variables. $S_{K2SN}$ is the solution obtained using the K2SN algorithm and $f(S_{K2SN})$ is the score for that solution. The next node to visit in the sequence is selected in the same manner as the application of ACS to the TSP. The local and global pheromone updating rules are still used and again similar to the TSP.

### 2.6.2 Bayesian Networks Using ACO - Searching in the space of DAGs

This approach is also guided by a colony of ants, but is carried out in the space of directed acyclic graphs (DAGs). The ant's behavior for this approach is based on a different Bayesian network leaning algorithm referred to as Ant B (Buntine, 1991). The representation of the problem domain is a graph, where the states of the problem are DAGs with *n* nodes. A state $G_h$ is a graph with nodes $x_i \in V$ with exactly *h* arcs and no directed cycle. The ant construction of the solution begins with the empty graph $G_0$ (arc-

less dag) and continues by adding an arc $x_j \rightarrow x_i$ to the current state $G_h$ in which the ant determines whether or not to halt the construction phase. The heuristic is to include in the graph the arc producing the greatest increase in the selected metric *f* (defined as):

$$\eta_{ij} = f\left(x_i, Pa\left(x_i\right) \cup \left\{x_j\right\}\right) - f\left(x_i, Pa\left(x_i\right)\right) \qquad [21]$$

The typical global and local pheromone update rules still apply. However, if $G^+$ is the best graph currently found, then $\Delta\tau_{ij}$ is computed as $\frac{1}{|(f(G^+ : D)|}$ if $x_j \rightarrow x_i \in G^+$. The pheromone is initialized as $\tau_0 = \frac{1}{n \cdot | f(G_{K2SN:D})|}$, where *n* is the number of variables and $G_{K2SN}$ is the network obtained by the K2SN heuristic. The next arc to be incorporated into the current graph *G*, is chosen by an ant using a stochastic decision rule taking into account the pheromone deposited at each arc.

### 2.6.3 Ant B Algorithm

The two main steps of this algorithm to build a solution consist of initialization and the iterative solution mechanism (deCampos, Fernandez-Luna, Gamez, & Puerta, 2002). Figure 4 outlines the Ant B pseudocode.

**//Initialization**
**for** $i=1$ **to** $n$ **do**:
  $Pa(x_i)=\varnothing$
**endfor**
**for** $i=1$ *and* $j=1$ **to** $n$ **do**:
  **if** $(i \neq j)$ **then**
    $\eta_{ij} = f(x_i, x_j) - f(x_i, \varnothing)$
  **end if**
**endfor**
**//Main loop**
**repeat**
  *Select two indices i and j by uing the probabilistic transition rule.*
  **if** $(\eta_{ij} > 0)$ **then**
    $Pa(x_i) = Pa(x_i) \cup \{x_j\}$
  **end if**
  $\eta_{ij} = -\infty$
  **for all** $x_a \in Ancestors(x_j) \cup \{x_j\}$ *and* $x_b \in Descendants(x_i) \cup \{x_j\}$ **do:**
    $\eta_{ab} = -\infty$
  **endfor**
  **for** $k=1$ **to** $n$ **do**:
    **if** $(\eta_{ik} > -\infty)$ **then**
      $\eta_{ik} = f(x_i, Pa(x_i) \cup \{x_k\}) - f(x_i, Pa(x_i))$
    **end if**
  **endfor**
  $\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \rho \cdot \tau_o$
**until** $\forall\, i, j\ (\eta_{ij} \leq 0\ or\ \eta_{ij} = -\infty)$.

**Figure 4. Ant B Algorithm (23)**

Ant colony optimization is applied in a similar manner to construct the network topology for the MCNDP. Arcs (edges) are chosen and added to the network graph using a probabilistic transition rule selecting the arc with the greatest potential.

## 2.7 Dynamic Environment

The related work to this point focused on global solutions to non-dynamic problems. The remainder of this chapter focuses on existing problems and solutions that include a dynamic environment. Two dynamic examples, the Dynamic Vehicle Routing Problem (DVRP) and the Dynamic Traveling Salesman Problem (DTSP), are discussed as both serve as a foundation for this research work.

### 2.7.1 Dynamic Vehicle Routing Problem (DVRP)

The Dynamic Vehicle Routing Problem (DVRP) has a number of common elements to the formulation of the network design problem (NDP). In the DVRP, multiple vehicles move through a graph and plan their routes based on known user priorities. Routes need to change dynamically according to changing user needs. For example, a new pickup request comes in part way during the day. The NDP is dynamic in the same way (user demands change over time), but has the additional dynamism in that the underlying delivery system (the network topology) also has the ability to change. Several solutions to the DVRP divide a day into time slices and solve each time slice entirely before moving on to the next (Montemanni, Gambardelaa, Rizzoli, & Donati, 2003) (Kilby, Prosser, & Shaw, 1998). To provide details into how these solutions work, a discussion of the vehicle routing problem is presented followed by an example.

The Vehicle Routing Problem (VRP) which forms the basis of the DVRP is a combinatorial optimization problem which is viewed as a merger of two problems: The Traveling Salesman Problem (TSP) and the Bin Packing Problem (BPP) (Machado,

Tavares, Pereira, & Costa, 2002). The Vehicle Routing Problem is NP-hard. The domain of VRPs has a large variety of problems including the dynamic VRP, capacitated VRP, multiple depot VRP, periodic VRP, split delivery VRP, stochastic VRP, VRP with backhauls, VRP with pick-up and delivery, VRP with satellite facilities, and VRP with time windows (Murata & Itai, 2005).

The general vehicle routing problem (VRP) is defined as a problem of designing routes for vehicles to service customers at known locations from and returning to a single starting location that minimizes the cost. Solving the VRP is compared to solving several travelling salesman problems (TSPs), one for each vehicle. All vehicles start at a depot and ultimately return to that same depot upon completion of their schedule route. Similar to the TSP, each location on a given route is only visited once.

The objective of the basic problem is to minimize a total cost as follows (Murata & Itai, 2005):

$$\min \sum_{m-1}^{M} c_m \tag{22}$$

where $M$ is the number of vehicles, each vehicle starts from the depot and is routed by a sequence of customers, then returns to the depot. The cost of each vehicle, $m \in M$, is denoted by $c_m$ and is computed as follows:

$$c_m = c_{0,1} + \sum_{i=1}^{n_m - 1} c_{i,i+1} + c_{n_m,0} \tag{23}$$

where $c_{i,j}$ is the cost to move between customers $i$ and $j$. The value of 0 is the index for the depot. Equation (23) states the sum of the cost between the depot and the first customer assigned to the *m-th* vehicle is identified as $c_{0,1}$, the total cost from the first customer to the $n_m$*-th* customer (the summation) and the cost between the final customer $n_m$ and the depot.

Equation 23 includes the cost from the depot to the first customer, the total cost, and the cost between the final customer and the depot. A complete description is provided in Table 1.

**Table 1. Vehicle Routing Problem (VRP) Cost Equation**

| Equation | Description |
|---|---|
| $c_{0,1}$ | cost from depot to first customer assigned to the *m*-th vehicle |
| $\sum_{i=1}^{n_m-1} c_{i,i+1}$ | total cost from the first customer to the $n_m$-th customer |
| $c_{n_m,0}$ | cost between the final customer $n_m$ and the depot |

Each vehicle is assigned to visit $n_m$ customers, which yields $N$ total customers in total and is computed as follows:

$$N = \sum_{m-1}^{M} n_m \qquad [24]$$

The model of the Vehicle Routing Problem (VRP) is a Graph, $G$ ($V$,$E$) where $V$ is set of vertices and $E$ is the set of edges. Details of the VRP are found in (Lenstra & K., 1981). Solutions to the VRP are found using the branch and bound technique (for up to 100 nodes) which computes every possible solution until one of the best is found (Fisher M. , 1994). Heuristic methods perform a limited exploration of the search space and typically obtain quality solutions within much shorter computation times (Kindervater & Savelsbergh, 1997). Finally, meta-heuristics perform a deep exploration of the most promising areas of the solution space and consist of simulated annealing, genetic algorithms, tabu search and ant algorithms.

The Dynamic Vehicle Routing Problem (DVRP) permits new orders to arrive throughout the working day after vehicles have already departed the depot (Montemanni, Gambardelaa, Rizzoli, & Donati, 2003). This new problem domain has evolved due primarily to advances in communication and information technologies such that information is being processed in real time. Orders change throughout the day and schedules need to be able to adjust quickly and dynamically.

To be able to adapt to this new environment, Montemanni discusses the use of an “event manager” that collects new orders, keeps a history of orders already served, and the current position of each vehicle. The event manager utilizes this information to develop a sequence of static VRP-like instances. The instances are then solved heuristically by an Ant Colony System (ACS) algorithm. The idea is to optimize a sequence of static VRP-like problems to produce a global solution that minimizes the travel time for each vehicle. The working day is divided into $n_{ts}$ time slots, each one long

$T_{ts} = T/\ n_{ts}$ seconds, where $T$ is the total length of the working day in seconds. Any new incoming order received during a given time slot is only considered at the end of that particular time slot. The purpose of using the time slot is to limit the amount of time dedicated to each static problem. This approach may be suitable to the NDP as also. The idea is to solve each "time slot" as a static NDP while minimizing the cost for that particular time slot. Once all the static NDPs are combined, this leads to a "global" solution that minimizes the total overall cost.

The paper presents an additional element called the pheromone evaporation procedure that passes information details of good solutions from the static VRP. After a time slot has completed and the associated static problem has been solved using the ACS algorithm, the pheromone matrix maintains information about good solutions. Good solutions generate higher values in the corresponding sections of the pheromone matrix (Montemanni, Gambardelaa, Rizzoli, & Donati, 2003). In the original ACS algorithm, the pheromone matrix τ is updated by only allowing the ant which constructed the best solution to reinforce the level of pheromone in the arcs that are part of the best solution, $S^+$, obtained so far. This directs the search in the neighborhood of the best solution. The updating rule is:

$$\tau_{ij} = (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij} \quad [25]$$

where $\rho$ is the pheromone evaporation parameter which controls the decay of the solutions over time. In the attached article, the "preferred route" is maintained in the pheromone trail matrix and future ants use this information to generate new solution based on this preferred route. *CostBest* is the total travel time of solution *BestSol*, the best

tour generated by the algorithm since computation began. The pheromone matrix is updated:

$$\tau_{ij} = (1-\rho)\,\tau_{ij} + \frac{\rho}{CostBest} \quad \forall (i,j) \in BestSol \tag{26}$$

The strategy to pass information on introduced a new parameter, $\gamma_r$ to regulate pheromone conservation (Montemanni, Gambardelaa, Rizzoli, & Donati, 2003). When a pair of “customers” exists in both the old and new static problem, the corresponding pheromone matrix entry is then initialized using the following formula:

$$\tau_{ij} = (1-\gamma_r)\,\tau_{ij}^{old} + \gamma_r \tau_0 \tag{27}$$

### 2.7.2 Dynamic Traveling Salesman Problem (DTSP)

The Traveling Salesman Problem (TSP) is one of the most intensely studied problems in computational mathematics. A formulation of the problem in terms of graph theory is: Given a complete weighted graph (where the vertices represent the cities, the edges represent the roads, and the weights are the cost or distance of that road), find a Hamiltonian cycle with the minimum weight (Michalewics & Fogel, 2004).

There are two variations of the Traveling Salesman Problem (TSP). The first, symmetric, is the case where traveling from city *X* to city *Y* has the same cost as traveling from city *Y* to city *X*. The second is asymmetric where the cost from city *X* to city *Y* might not necessarily be the same as traveling from city *Y* to city *X*.

For years, the TSP has served as a “platform” for investigating optimization techniques. The TSP belongs to the class of NP-complete problems. Finding optimum solutions for NP-complete problems increase faster than polynomial with the problem

size $N$. The most direct solution is to try all the permutations (ordered combinations) and identify the one that is cheapest (using a brute force search approach). However, given that the number of permutations is $n!$ (with $n$ being the number of cities), this solution quickly becomes impractical (Michalewics & Fogel, 2004). The cost for the symmetric TSP reduces the search space by one-half since proceeding down the list of cities from right to left or left to right is the same. Consequently, the actual search space for the symmetric TSP reduces to $(n-1)!/2$.

A Dynamic Traveling Salesman Problem (DTSP) is much more complicated than a general TSP. The DTSP must handle aspects that exist in real world situations such as the number of cities changing and the travel cost between cities to change. Therefore, a dynamic traveling salesman problem has some additional properties (Zhou, Kang, & Yan, 2003). First, the number of cities may change over time, i.e., new cities may be added while existing cities may be deleted. Second, the city locations may continuously change with time, modifying the distance or cost between cities. A unified dynamic cost (or distance) matrix is:

$$D(t) = \{d_{ij}(t)\}_{n(t)\mathbf{x}(nt)} \qquad [28]$$

where $d_{ij}(t)$ is the cost to travel from city $i$ to city $j$ at time $t$, $n(t)$ is the number of cities at time $t$, and $t$ is the real world time. A formal definition of the DTSP is:

Given $n(t)$ cities and a dynamic distance matrix $D(t) = \{d_{i,j}(t)\}_{n(t)\mathrm{x}n(t)}$ , minimize:

$$d(\pi(t)) = \sum_{j=1}^{n(t)} d_{\pi_j \pi_{j+1}}(t) \qquad [29]$$

where $\pi(t)$ is a permutation over the set $\{1, 2 \ldots n(t)\}$ and $\pi_{n(t)+1} = \pi_{1.}$

Eyckelhof and Snoek present an Ant System approach to a dynamic Traveling Salesman Problem (Eyckelhof & Snoek, 2002). The authors propose a variation on the TSP to be used with the Dynamic Traveling Salesman Problem (DTSP). They introduce a "traffic jam" on a particular road which is associated with an increase in cost (or more precisely travel time) between two cities.

A lower boundary on the amount of pheromone used on every road is introduced into the AS-DTSP. This technique is called "shaking" the environment to smooth out all pheromone levels. In a static environment, high amounts of pheromone on a given road typically ensure that road is selected. This works fine for the static problem. However, in a dynamic environment, it prevents taking an alternative path when a traffic jam occurs. Shaking modifies the ratio between the amounts of pheromone on all roads, while still preserving the relative ordering. Global shaking leads to potential loss of information. Additional modifications included the levels of pheromone. After pheromone is evaporated, the amount of pheromone does not drop below a predetermined level. Also, shaking is used as a smoothing operator to the pheromone matrix. When the "shaking percentage", $p$ is set to 1, the operator acts globally. When $p$ is between zero and one, local shaking occurs. When $p$ is equal to zero, there is no shaking.

Guntsch presents an "Ant Colony Optimization (ACO) approach for a Dynamic Traveling Salesperson Problem (DTSP)" (Guntsch, Middendorf, & Schmeck, 2001). With the dynamic aspect, cities are either inserted or deleted over time. In their approach, selected cities are replaced with new cities at varying frequencies. In addition, the authors studied pheromone modification with respect to the dynamic changes being introduced into the problem domain. The strategies are based upon the degree of locality associated with the inserted/deleted cities and whether or not a modified elitist ant is kept or discarded.

It was found that information resets by equalizing the pheromone values, which influence the level of experience the ants may use to construct a solution. The authors present three basic strategies. Each strategy distributes a reset-value to each city which is used to compute the change to the pheromone values on edges incident to a particular city.

## 2.8 Distributed Environment Optimization Solution Techniques

Beyond handling change within a highly dynamic network environment, algorithms must be able to respond quickly. Distributed solutions are becoming more prevalent as a means to reduce computation time as compared to centralized solvers. This section discusses distributed solutions with respect to network topology, the Vehicle Routing Problem (VRP), Traveling Salesman Problem (TSP), and fault-tolerant routing for large scale distributed parallel computing.

### 2.8.1 Distributed Network Topology

Designing proper routing schemes for effective communication between any source and destination within a wireless ad hoc network is a challenge. In the operation of a network, some nodes may enter or depart randomly, some links among nodes may be constructed or removed, and the distances between nodes may change. Dynamic network topology modifications are critical to ensure network connectivity (Jiang & Jiang, 2005). Conventional routing protocols and network design problem solutions require knowledge of the entire network and are not suitable in a strong dynamic environment primarily because the network topology needs to be updated throughout the network (Choudhury, Bandyopadhyay, & Paul, 2000).

Choudhury discussed a mobile multi-agent based framework to address the topic of topology discovery in ad hoc wireless network environments (Choudhury, Bandyopadhyay, & Paul, 2000). Their goal is to collect all topology-related information from each individual node in the network and then distribute the data periodically in updates to other nodes using mobile agents. A predictive algorithm running on each node predicts the current network topology based on the available network information stored at that particular node. They demonstrate via simulation the concept of a mobile multi-agent framework that makes each node in the network aware of the topology with minimal network overhead. In addition, incorporating topology information into the nodes ensured distributed network management and implemented communication awareness automatically. The cooperation of the nodes is achieved in terms of social

rules. Given each node has the same topology information and uses the same algorithm, each node reaches the same solution that they then implement.

Mobile agents or messengers navigating throughout the network are an effective solution to the problem of topology discovery (Choudhury, Bandyopadhyay, & Paul, 2000). The agents move from node to node, collect information from these nodes, communicate with other agents collecting updates of portions of the network they have not visited, and share this new data with newly visited nodes and agents. The migration time interval is controlled to control the agent traffic in the network to prevent the network from being flooded with propagation of agents. Also, an agent only migrates from a particular node to a neighbor node to ensure the network does not get flooded with agent propagation. A critical aspect of this topology information is the degree of correctness concerning the information being carried throughout the network. An important concept discussed is recency of information. Each node maintains a counter called recency token and the node with the higher recency token is deemed to have more current information.

Another method to solve the network programming problem used iterative execution of the same algorithm at each node of the network (Dennis, 1964). For any given node, the algorithm only needs data from an adjacent node on the network. He implemented the methodology through the use of electrical models and the techniques he developed are shown to be useful for message routing and similar problem domains previously relying on computation performed at a global level.

### 2.8.2 Distributed Vehicle Routing Problem

The current policy concerning logistics planning is to execute decision-making at a central authority (Toth & Vigo, 2001). This approach results in solving large optimization problems in order to find the optimal solution with respect to vehicle loads and routes (Wenning, Timm-Giel, & Pesch, 2006). The optimization problems are typically of a static nature with fixed assumptions concerning transport needs and the traffic situation. However, when dynamic changes occur, such as unexpected transport orders or changes in the road environment, the optimization becomes even more complex. The dynamism and the distribution that is characteristic of real world vehicle routing problems have led to recently developed techniques to solve the vehicle routing problem in a distributed manner. This shift from centralized solvers to decentralized solvers is facilitated by the availability of information and communication technologies. One particular approach is investigating the effects of autonomous decision making by each individual transport vehicle with respect to the route and loading it believes is happening. The approach makes the assumption that routing decisions and the choice of packages are carried out by the vehicles themselves. The methodology uses a reward function based on the load and route choices to optimize network performance (Wenning, Timm-Giel, & Pesch, 2006).

Other techniques have been based upon multi-agent approaches (Bachem, Hochstattle, & Malich, 1993) (Fisher & Kuhn, 1993) (Burckert, Fischer, & Vierke, 1998). A specific multi-agent model based on coalition formation and dedicated to the VRP with Time Windows (VRPTW) is called Coal-VRP (Kefi & Ghedira, 2004). Coal-

VRP determined a final set of routes given a set of orders received from geographically distributed customers. There are two basic types of agents, the Interface agent and the Customer agent. Although this model showed interesting and promising results, its disadvantages are spatial and temporal complexity, restricting its applicability to real world problems. However, a new model was developed called the Dynamic generation of Coalition for solving a VRP (DyCoal-VRP) (Boudali, W., & Ghedira, 2004). This model eliminated the coalition list and introduced a dynamic generation of coalitions. The new solving process is carried out in only two steps which are a distributed formation of the relationships graph and the construction of the most desired coalition for each Customer agent. They later adopt an approach which is based on interaction between agents and the objective is to dynamically generate collisions where each coalition directly corresponds to a vehicle's route (Boudali, Fki, & Ghedira, 2005).

### 2.8.3 Distributed Traveling Salesman Problem

In 2007, a protocol was introduced for a local search and a genetic algorithm for solving the distributed traveling salesman problem (TSP) (Sakuma & Kobayashi, 2007). For the distributed TSP problem domain, information concerning the cost function to include traveling costs between cities and cities still under consideration to be visited are separated by distributed parties and are kept private from each other. They further discuss a protocol which solves the distributed TSP in a secure manner using a combination of genetic algorithms and a cryptographic technique referred to as the secure multiparty computation.

The Chained Lin-Kernighan (CLK) algorithm has been shown to be a very effective heuristic for solving the Traveling Salesman Problem (TSP) (Fisher & Merz, 2005). The proposed algorithm is distributed where each network node uses the CLK algorithm to locally optimize TSP instances. They compare the distributed approach with the original CLK algorithm. Given the same total amount of computational time, the distributed methodology outperformed the original CLK by finding better tours.

### 2.8.4 FIRE Ant

An algorithm titled Fault-Tolerant Intelligent Routing Environment AntNet (FIRE Ant) is the first agent algorithm developed to handle multiple network failures or disaster situations (Thomas, 2004). Fire Ant permits a network to evade network disasters by using an agent-based decentralized routing methodology. FIRE Ant is essentially a cost routing approach which optimizes the average network flow by balancing the data flow during situations of network disaster.  The research demonstrates a fault-tolerant routing algorithm designed on the concept of large-scale distributed parallel processing. Communication is able to continue to occur even while the network is experiencing faults. Agents are provided the necessary information to make their own adaptive routing decisions based upon network performance and decentralized knowledge of network failures.

## 2.9 Summary

Chapter II provides current information on network topology control and presents the Multi-commodity Capacitated Network Design Problem (MCNDP). Several methodologies to solve the MCNDP are presented: Mixed Integer Linear Program (MILP), Network Flow Algorithms, Genetic Algorithms (GAs), and the Tabu Search algorithm. Then, the Ant Colony Optimization (ACO) algorithm is presented, followed by an application of ACO with Bayesian Networks. Next, applications with respect to dynamic environments, such as the Dynamic Vehicle Routing Problem (DVRP) and Dynamic Traveling Salesman Problem (DTSP) are discussed. Lastly, distributed processing is addressed with respect to network design, the Traveling Salesman Problem (TSP) and the Vehicle Routing Problem (VRP). These particular topics are of interest as similar approaches are incorporated into the development of the proposed research methods.

## III. Methodology

Connecting computers in a network consisting of high-bandwidth directional links and determining the optimal routing of information in a highly dynamic network environment is challenging. This research presents a novel approach to solving the Multi-commodity Capacitated Network Design Problem (MCNDP) using the Ant Colony System (ACS) to construct the network topology. Two applications of ACS to the MCNDP problem are Ant Colony System Standard (ACSS) and Ant Colony System Estimation (ACSE). These two approaches differ in how the value of a solution is calculated. Additionally, these methods are extended to finding solutions under strong dynamic conditions, and instead of being centralized solvers are distributed in the Distributed Ant Colony System Standard (DACSS) and Distributed Ant Colony System Estimation (DACSE).

### 3.1 Overview

The learning of the network for the MCNDP problem using ACS uses a similar technique as learning a Bayesian Network (deCampos, Gamez, & Puerta, 2002). In both, the data structure being learned is a graph, in the case of the MCNDP this is the network. A network problem for which the network must be built contains a list of nodes and a list of commodities. In addition, each node object contains a list of regular edges and potential edges. The list of regular edges used in the topology is empty (the network starts with no links active).

Each ant in building its solution selects an edge from the list of all potential edges. The selected potential edges are then transformed into regular edges and form the

network topology. The ants iteratively stochastically select edges based on the pheromone on each edge, and the heuristic evaluation of the edge. Once all potential edges have been explored, the ant stops searching.

The initial application of Ant Colony Optimization (ACO) to solve the MCNDP shows the algorithm is a viable solution method for this problem. Specifically, two ACO applications to the MCNDP are developed, ACSS and ACSE. ACSS uses an actual routing strategy to evaluate the overall cost for each ant solution generated. ACSE incorporates heuristic information to evaluate the ant solutions, eliminating the need to route the commodities for each ant after an iteration is completed. These two new approaches are comparatively tested against the previous solutions, Mixed Integer Linear Program (MILP) (Erwin, 2006) and Network Flows (Garner, 2007). These two algorithms are then incorporated into a highly dynamic network environment.

The dynamic aspect added to the Network Design Problem (NDP) includes changes occurring to and within the network. Links (edges) which currently exist between nodes become suddenly unavailable. The fixed edge costs of an edge changes. The cost to route commodities on particular edges are modified. Links between nodes are turned on and off during algorithm execution. The new problem is now of a dynamic nature and the topology solver needs to be able to adapt to all of these changes and still produce an approximately optimal solution. Three categories of the ACSS and ACSE approaches are created. The network is monitored at various levels of change, ranging from none (static network) to a highly dynamic (50%) change occurring to the network.

Lastly, the solution methodology is transformed from requiring a centralized global solver to a distributed solver. In the distributed solver, each node runs its own

ACS algorithm (ACSS and ACSE) which is used to select the edges used to form the network topology. Once a node has chosen its edges, the selected edge list becomes available to the Network Event Manager. After the Network Event Manager receives each node's selected edge list, the overall network topology is constructed and provided to the appropriate routing algorithm at the global level. The network routing occurs at the global level to provide a consistent means to evaluate the constructed network topology. As changes occur within the network (existing links become unavailable, new links become available, change in costs, etc) within the network, the Event Manager informs each node to make updates to its selected edge list. Each node then acts independently and using ACS or ACSE, updates its list. This enhancement dramatically decreases computation time while still developing comparable network topology solutions to the centralized approach.

### 3.1.1 Ant Colony System Standard (ACSS)

The first application of ACS to the MCNDP is the ACSS. The ACSS builds the network topology in a similar manner as in learning a Bayesian Network. The evaluation of the built network topology is conducted with a full commodity routing and evaluation of the network. The routing of commodities provides the objective score of the proposed network. ACSS uses a centralized solver approach for evaluating all the potential edges and has global knowledge of the entire network. Each node's outgoing potential edge list is consolidated into one combined list for the network. Each node has knowledge of which of its edges are being selected for the overall network topology until a final solution is generated. Figure 5 shows how this process flows.

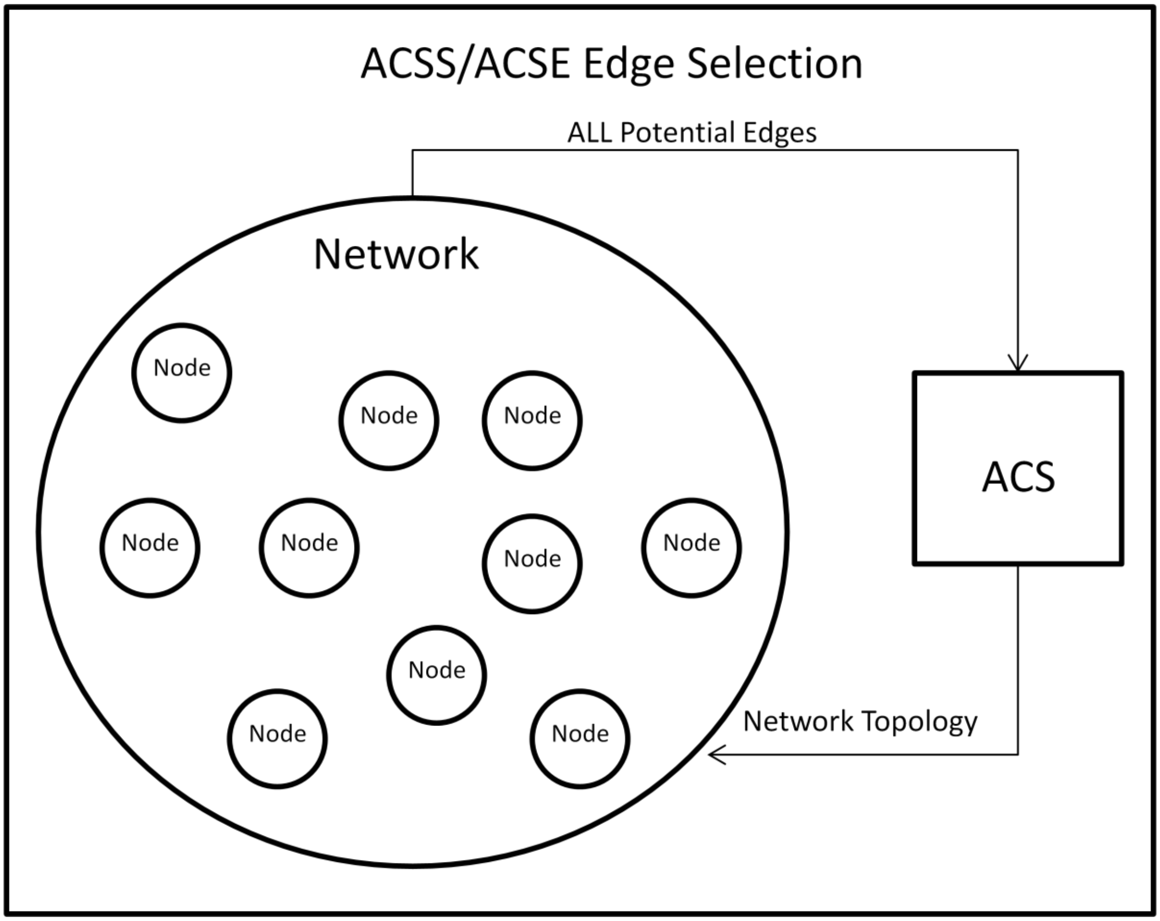


**Figure 5. ACSS/ACSE Edge Selection**

ACSS produces near optimal network costs when compared to the MILP (Erwin, 2006) and Network Flow (Garner, 2007) solutions. Although the algorithm produces desirable solutions, an area of concern is the amount of time required to obtain the solution. The components of the ACSS algorithm are as follows:

- Problem representation is the entire network graph. Let $N$ denote the set of nodes, $K$ the number of commodities, $F$ the number of interface types, and G the number of groups. Further, let $e_{ijf}$ denote the edge connecting node $i$ to

node $j$ by the interface type $f$. The search space is all of each node's set of potential outgoing edges, $e_{ijf} \in E$. The solution space $S$, consists of all potential outgoing edges, $e_{ijf}$ selected from $E$ as outgoing edges, $s_{ijf}$. The desirable solution $S$ is the one with the minimum total cost. The fixed edge cost is $c_{ijf}$ and the variable flow cost is denoted $v_{ijf}^{k}$, which is the per unit cost of commodity $k$ on edge $e_{ijf}$. If $m^k = 1$, then commodity $k$ is dropped. The penalty function is 1000 for each dropped commodity multiplied by the commodities demand. The demand of a commodity is denoted $d_{ijf}^{k}$, the demand of commodity $k$ on $e_{ijf}$. Also, a group is defined as a subset of $E$, $G \subset E$ and is the set of all potential edges, $e_{ijf}$ from a specific node $i$ sharing the same interface type $f$. Furthermore, only one potential edge $e_{ijf} \in G_{if}$ is able to be selected from within a group to be added to the solution set $S$.

- The heuristic calculation for ACSS is similar to the calculation for the TSP. In the TSP, $\eta_{ij} = 1/d_{ij}$, where $d_{ij}$ is the distance between cities $i$ and $j$. For the MCNDP,

$$\eta_{ijf} = \frac{1}{c_{ijf}} \qquad [30]$$

where $c_{ijf}$ is the fixed edge cost of including edge $e_{ijf}$ in the network.

- The evaluation criteria for the network topology routes all commodities through the network using one of the greedy network flow routing algorithms (Garner, 2007) to compute the overall cost. The total cost of the solution consists of the sum of all fixed edges costs, all variable flow cost and any

penalties associated with dropped commodities. The total fixed edge cost, $c_{total}$ is computed as:

$$c_{total} = \sum c_{ijf} \ \forall s_{ijf} \in S \qquad [31]$$

The variable flow cost, denoted $v_{total}$ is:

$$v_{total} = \sum v_{ijf}^{k} \ \forall s_{ijf}^{k} \in S \qquad [32]$$

The penalty $p^k$ associated with any dropped commodities is:

$$p^k = 1000(m^k)(d_{ijf}^{k}) \qquad [33]$$

Therefore, the total cost of the network topology is $c_{network}$ and is computed by:

$$c_{network} = c_{total} + v_{total} + \sum_{i=1}^{n(n-1)} p^k \qquad [34]$$

where $n$ is the total number of nodes in the network.

- Pheromone is initialized by depositing a small amount of pheromone, $\tau_0$ in every potential edge of the graph. This initial amount is calculated by finding the network cost using Garner's greedy approach and the Edmonds Karp, best-first search (Garner, 2007). The initial pheromone value is

$$\tau_0 = \frac{1}{c_{network}} \qquad [35]$$

which is the inverse of the total network cost.

- The probabilistic transition rule is similar to the one used for the TSP. Pheromone is modeled by means of a matrix $\tau$, where $\tau_{ijk}$ contains the level of pheromone deposited in the edge from node $i$ to node $j$ on interface $k$. An ant

$a$ selects the next available edge to add to the network topology with probability:

$$p_{ijf}^{a} = \begin{cases} \dfrac{\left[\tau_{ijf}\right]^{\alpha}\left[\eta_{ijf}\right]^{\beta}}{\sum_{e_{ijf} \in E^{a}}\left[\tau_{e_{ijf}}\right]^{\alpha}\left[\eta_{e_{ijf}}\right]^{\beta}} & \text{if } ijf \in E \\ 0 & \text{otherwise} \end{cases} \qquad [36]$$

where $\eta_{ijk}$ represents heuristic information about the problem; $E$ is the set of all available potential edges; $\alpha$ and $\beta$ are parameters that determine the relative importance of the pheromone with respect to the heuristic information.

- The local update rule, which updates the pheromone for every edge selected is:

$$\tau_{ijf} = (1-\xi)\tau_{ijf} + \xi\tau_0,\ 0.0 < \xi < 1.0 \qquad [37]$$

where $\xi$ is the evaporation parameter.

- The global update rule, which updates the pheromone for those edges that are part of the best solution is:

$$\tau_{ijf} = (1-\rho)\tau_{ijf} + \rho\Delta\tau_{ijf},\ 0.0 < \rho < 1.0 \qquad [38]$$

where $\rho$ is the evaporation parameter.

The pseudocode for the ACSS algorithm is shown in Figure 6.

**// Initialization**
**for** each node $n_i \in N$ **do**
    **for** each potential edge $e_{ijf}$ in $n_i$ **do**
        insert $e_{ijf}$ into potential edge list $E$
    **end for**
**end for**
*route commodities using EdmondKarp1, greedy selection*
$\tau_0 = \frac{1}{c_{network}}$
**for** each edge $e_{ijf} \in E$ **do**
    $\tau_{ijf} = \tau_0$
**end for**
//**Main loop**
**for** total number of iterations **do**
    **for** each ant *a* **do**
        **for** each potential edge $e_{ijf} \in E$ **do**
            $e_{ijf} = visited(FALSE)$
        **end for**
        **while** $\exists\, e_{ijf} \in E = visited(FALSE)$ **do**
            $q = rand[0,1]$
            **// Get transition probabilities**
            **for** each potential edge $e_{ijf} \in E$ **do**

$$p^a_{ijf} = \begin{cases} \dfrac{\left[\tau_{ijf}\right]^{\alpha}\left[\eta_{ijf}\right]^{\beta}}{\sum_{e_{ijf} \in E^a}\left[\tau_{e_{ijf}}\right]^{\alpha}\left[\eta_{e_{ijf}}\right]^{\beta}} & \text{if } ijf \in E \\ 0 & \text{otherwise} \end{cases}$$

            **end for**
            **//Make edge selection**
            **if** $(q \le q_0)$ **then**
                $s_{ijf}$ = $\min(p^a_{ijf}, e_{ijf} \in E)$
            **else**
                *randomly select edge* $s_{ijf}, e_{ijf} \in E$
            **end if**
            **//All edges in selected group, visited is true**
            $\forall\, e_{ijf} \in E_G = visited(TRUE)$
            **// Update local pheromone**
            $\tau_{ijf} = 1-\xi \;\; \tau_{ijf} + \xi\tau_0$
        **end while**

*evaluate solution using a network flow algorithm*
**if** (*currentScore* < *bestScore*) **then**
*bestScore* = *currentScore*
*bestNetwork* = *currentNetwork*
**end if**
**end for**
**// Update global pheromone for best ant solution**
**for** each $s_{ijf} \in S$
$\tau_{ijf} = 1-\rho \;\; \tau_{ijf} + \rho\Delta\tau_{ijf}$
**end for**
**end for**

**Figure 6. Pseudocode for the ACSS Algorithm**

### 3.1.1 Ant Colony System Estimation (ACSE)

The second ACS approach, ACSE, uses several approximate heuristics in place of the fixed edge cost heuristic $c_{ijf}$ used in ACSS and in place of performing a full network routing. The use of the heuristics to evaluate the network solution eliminates the need to route commodities repeatedly through the network using the Edmonds Karp greedy selection algorithm. In ACSE, routing occurs only one time at the end of the process to evaluate the network's objective score, drastically reducing the time required to produce a network topology solution.

Several combinations of the weighted cost heuristics were developed. Table 2 lists the 14 test cases identifying the percentage of the fixed edge cost, variable edge cost, edge capacity, and edge value for the transition probability rule and solution evaluation criteria.

**Table 2. ACSE Heuristics**

| | Transition Probability Rule | | | | Evaluation Criteria | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Test Case | Fixed Edge Cost ($p_1$) | Variable Edge Cost ($p_2$) | Edge Capacity ($p_2$) | Edge Value ($p_2$) | Test Case | Fixed Edge Cost ($p_1$) | Variable Edge Cost ($p_2$) | Edge Capacity ($p_2$) | Edge Value ($p_2$) |
| 1 | 50% | 50% | -- | -- | 1 | 50% | 50% | -- | -- |
| 2 | 100% | 0% | -- | -- | 2 | 100% | 0% | -- | -- |
| 3 | 80% | -- | 20% | -- | 3 | 80% | -- | 20% | -- |
| 4 | 90% | -- | 10% | -- | 4 | 90% | -- | 10% | -- |
| 5 | 95% | -- | 5% | -- | 5 | 95% | -- | 5% | -- |
| 6 | 80% | -- | 20% | -- | 6 | 100% | -- | 0% | -- |
| 7 | 90% | -- | 10% | -- | 7 | 100% | -- | 0% | -- |
| 8 | 95% | -- | 5% | -- | 8 | 100% | -- | 0% | -- |
| 9 | 80% | -- | -- | 20% | 9 | 80% | -- | -- | 20% |
| 10 | 90% | -- | -- | 10% | 10 | 90% | -- | -- | 10% |
| 11 | 95% | -- | -- | 5% | 11 | 95% | -- | -- | 5% |
| 12 | 80% | -- | -- | 20% | 12 | 100% | -- | -- | 0% |
| 13 | 90% | -- | -- | 10% | 13 | 100% | -- | -- | 0% |
| 14 | 95% | -- | -- | 5% | 14 | 100% | -- | -- | 0% |

The ACSE algorithm was evaluated using the 14 test cases identified in Table 2. For each test identified in Table 2, the algorithm uses the percentages indicated for the parameters *p1* and *p2* in the various heuristics discussed below. Using a heuristic approximation for the evaluation criteria eliminates the need to actually route the commodities as the evaluation function, significantly reducing computation time.

The components of the ACSE algorithm are similar to the ACSS algorithm but with the following modifications:

- Problem representation and search space is same as ACSS.
- The following ACSE heuristics and heuristic combinations are used:

1. The fixed edge heuristic is:

$$\eta_{ijf} = \frac{1}{c_{ijf}} \quad [39]$$

where $c_{ijf}$ and is the fixed edge cost from node $i$ to node $j$ using interface $f$. This equation corresponds to Test Case 2 in Table 2.

2. The sum of the fixed edge cost and the average variable flow cost*:*

$$v_{total} = \sum_{k=1}^{n(n-1)} v_{ijf}^{k} \quad [40]$$

where $v_{ijf}^{k}$ is the per unit cost of commodity $k$ on edge $e_{ijf}$ and n is the number of nodes. The average variable flow cost for edge $e_{ijf}$ *is*:

$$v_{ijf} = \frac{v_{total}}{n(n-1)} \quad [41]$$

The sum of the fixed edge cost and the average variable flow cost heuristic is:

$$\eta_{ijf} = \frac{1}{(p_1)c_{ijf} + (p_2)v_{ijf}} \quad [42]$$

The values of $p_1$ and $p_2$ are identified in Table 2 and Test Case 1 uses this equation.

3. Weighted sum of fixed cost and edge capacity heuristic:

$$\eta_{ijf} = (p_1)\frac{1}{c_{ijf}} + (p_2)e_{ijf}^{c} \quad [43]$$

where $p_1 + p_2 = 1.0$ and $e_{ijf}^{c}$ is the capacity of an edge from node $i$ to node $j$ using interface $f$. The actual values for $p_1$ and $p_2$ are listed in Table 2. Equation 43 is utilized in Test Cases 3, 4, 5, 6, 7, and 8.

4. Weighted sum of fixed cost and the average edge value:

$$e_{total}^{v} = \sum_{k=1}^{n(n-1)} d_{ijf}^{k} \qquad [44]$$

where $d_{ijf}^{k}$ is the demand of commodity $k$ from node $i$ to node $j$ using interface $k$ and $e_{total}^{v}$ is the total demand of all commodities using the edge $e_{ijf}$

$$e_{ijf}^{v} = \frac{e_{total}^{v}}{n(n-1)} \qquad [45]$$

where $n(n-1)$ is the total number of commodities and $e_{ijf}^{v}$ is the average demand for the edge $e_{ijf}$. The weighted sum of the fixed cost and the average edge value heuristic is:

$$\eta_{ijf} = (p_1)\frac{1}{c_{ijf}} + (p_2)e_{ijf}^{v} \qquad [46]$$

where $p_1 + p_2 = 1.0$. Again, the actual values for $p_1$ and $p_2$ are listed in Table 2 and this equation is valid for Test Cases 9, 10, 11, 12, 13, and 14.

- The evaluation criteria for the proposed network topology use one of the heuristic values identified above with the corresponding percentages ($p_1$ and $p_2$) listed in Table 2 to estimate the routing of all commodities through the network topology. However, once a network topology is selected, the final network score is constructed using the specified network flow routing algorithm (Garner, 2007) to include the fixed edge cost, the commodity flow cost and any penalties associated with dropped commodities (same as ACSS - see Equation 34).

- Pheromone is initialized by depositing a small amount of pheromone, $\tau_0$ in every potential edge of the graph. This initial amount is calculated by finding the minimum cost edge within a group, taking the sum of each group's minimum cost edge, and then using the inverse of the sum. A group is a set of edges that belong to the same source node and are of the same interface. The elements in the set are the destination nodes that could potentially be connected using this particular interface.

$$\tau_0 = \frac{1}{\left(\sum_{i=1}^{n}\sum_{f=1}^{3}\min(c_{ijf}), \forall e_{ijf} \in G_{if}\right)} \qquad [47]$$

- The probabilistic transition rule is the same as ACSS (Equation 36). However, there are several variations using the heuristics above and the percentages for $p_1$ and $p_2$ identified in Table 2.
- The local update rule, with $\xi$ being the evaporation parameter, is the same as ACSS (Equation 37).
- The global update rule, with $\rho$ being the evaporation parameter, is also the same as ACSS (Equation 38).

The pseudocode for the ACSE algorithm is presented in Figure 7.

**// Initialization**
**for** each node $n_i \in N$ **do**
    **for** each potential edge $e_{ijf}$ in $n_i$ **do**
    insert $e_{ijf}$ into potential edge list $E$
    **end for**
**end for**
$\tau_0 = \dfrac{1}{\left(\sum_{i=1}^{n}\sum_{f=1}^{3} min(c_{ijk})\forall j\right)}$
**for** each edge $e_{ijf} \in E$ **do**
    $\tau_{ijf} = \tau_0$
**end for**
//**Main loop**
**for** total number of iterations **do**
    **for** each ant $a$ **do**
        **for** each potential edge $e_{ijf} \in E$ **do**
            $e_{ijf} = visited(FALSE)$
        **end for**
        **while** $\exists\, e_{ijf} \in E = visited(FALSE)$ **do**
            $q = rand[0,1]$
            **// Get transition probabilities**
            **for** each potential edge $e_{ijf} \in E$ **do**
$$p_{ijf}^{a} = \begin{cases} \dfrac{\left[\tau_{ijf}\right]^{\alpha}\left[\eta_{ijf}\right]^{\beta}}{\sum_{e_{ijf}\in E^{a}}\left[\tau_{e_{ijf}}\right]^{\alpha}\left[\eta_{e_{ijf}}\right]^{\beta}} & \text{if } ijf \in E \\ 0 & \text{otherwise} \end{cases}$$
            **end for**
            **//Make edge selection**
            **if** $(q \le q_0)$ **then**
                $s_{ijf}$ = $\min(p_{ijf}^{a}, e_{ijf} \in E)$
            **else**
                *randomly select edge* $s_{ijf}, e_{ijf} \in E$
            **end if**
            **//All edges in selected group, visited is true**
            $\forall\, e_{ijf} \in E^{g} = visited(TRUE)$
            **// Update local pheromone**
            $\tau_{ijf} = 1-\xi \;\; \tau_{ijf} + \xi\tau_0$
        **end while**

        *evaluate solution using a heuristic method*
        **if** (*currentScore* < *bestScore*) **then**
            *bestScore* = *currentScore*
            *bestNetwork* = *currentNetwork*
        **end if**
    **end for**
    **//Update global pheromone for best ant solution**
    **for** each $s_{ijf} \in S$
        $\tau_{ijf} = 1-\rho \;\; \tau_{ijf} + \rho\Delta\tau_{ijf}$
    **end for**
**end for**

**Figure 7. Pseudocode for the ACSE Algorithm**

### 3.1.3 Dynamic MCNDP (DMCNDP)

As the network experiences both weak and strong dynamics, the problem domain shifts from a static to a highly dynamic environment. The current topology may no longer present a feasible solution. Therefore, the topology of the network must adapt. For a gradual change (weak dynamics), the actual change is transparent. Weak dynamics occurring within the network are modest changes (0 to 10 percent modifications) to the fixed edge costs, variable flow costs, edge availability, edge capacity, commodity demand, and commodity value. Whereas strong dynamics occurring within the network consist of similar types of changes, but at a much larger scale of change (25 percent or more). Topology construction algorithms need to adapt to these changes and still produce quality solutions.

Three dynamic categories were developed to compare and contrast how the alterations to the ACS algorithm to account for strong domain dynamics performed. Table 3 provides a description of the three categories.

**Table 3. Dynamic Algorithm Categories**

| CATEGORY | NAME | DESCRIPTION |
|---|---|---|
| **1** | **Baseline** | Baseline algorithm with no additional changes or modifications. This algorithm is used to determine how the existing ACS algorithm handles dynamic change within the network environment |
| **2** | **Restart** | Restart algorithm knows exactly when a change occurs and reinitializes the pheromone matrix after each pre-set interval of 25 iterations. Again, this algorithm is used for comparison testing. |
| **3** | **Dynamic** | Dynamic algorithm dynamically adapts to the network environment. The Dynamic algorithm modifies the $q_0$ parameter based upon the convergence of the solutions found by the previous ants during exploration. |

During an iteration of both ACSS and ACSE, each individual ant constructs its own network solution. During the ants' traversal of the graph, they lay pheromone, and the pheromone on the edge also evaporates. A significant feature of the ACS algorithm is the flexibility to tradeoff between exploration and exploitation through the value assigned to the exploration parameter $q_0$ which is a real number between 0 and 1. When $q_0$ is set to 1, ants greedily select the path with the most pheromone (exploit). When $q_0$ is equal to 0, the ants randomly select edges (explore). The value of $q_0$ is set to 0.9 for both the Baseline and Restart categories (Dorigo & Stutzle, 2004). However, the Dynamic category includes an ACS alteration that dynamically adjusts the value of $q_0$ as the network experiences change.

The dynamic algorithm modification analyzes the scores of each ant generation. After an ant generation, the ants' scores are stored. The values from the five most recent generations then become input to a probability density function and the returned values are then averaged. This mean value is then used to calculate a new updated $q_0$ value for the modified algorithm. Therefore, the value of $q_0$ increases, decreases, or remains the same based on the exploration and exploitation of the five most recent ant generations. This feature enables these modified algorithms to better anticipate change to the network and adapt more quickly.

Preliminary research studied the Dynamic Traveling Salesman Problem (DTSP) in a dynamic environment and incorporated this new approach. . Extensive testing was conducted using these three categories of ACSS and ACSE for the DTSP to determine the appropriate probability density function to incorporate into the Dynamic algorithm. Initially, the normal distribution and Inverse Gaussian distribution were implemented as potential probability density functions to map the ant scores to.

For each approach, changes were made to the distance matrix at iteration 250, 500, and 750. There were a total of five different changes to the various paths of the distance matrix. All path changes were randomly selected and either involved an increase, decrease or no change to the distances of paths found on the current best solution.

The first category, Baseline, had no changes introduced except to the distance matrix. Restart reinitialized the pheromone matrix at each interval (250, 500, and 750). The Dynamic category modified the $q_0$ parameter based upon the convergence of the

solutions found by the previous 10 ants during exploration. The parameter settings for all three algorithms were set as recommended in (Dorigo & Stutzle, 2004).

A comparative analysis between the three algorithms was conducted using four test files. The four files are ulyssess22, berlin52, eil101, and kroA100 from the TSPLIB, a library of sample problems for the traveling salesman problem. For all four tests, the algorithms were run 30 times. During each run, the algorithms conducted 1000 iterations. After reviewing test results, the Inverse Gaussian distribution produced the overall best results and was selected as the best function for the distribution calculation. The probability density function of the Inverse Gaussian distribution (Chhikara & Folks, 1989) is:

$$f(x;\mu,\lambda) = \left[\frac{\lambda}{2\pi x^3}\right]^{\frac{1}{2}} exp\,\frac{-\lambda(x-\mu)^2}{2\mu^2 x} \quad for\ x > 0 \qquad [48]$$

ACSS Dynamic (ACSS-D) and ACSE Dynamic (ACSE-D) for the DMCNDP also include this feature. Both algorithms are capable of adjusting the value of the $q_0$ parameter based upon the convergence of the solutions found by the ants during exploration. The value of $q_0$ is updated based upon the ants' scores found after each ant generation. The pseudocode for this algorithm is shown in Figure 8.

```
// Initialization
//This algorithm used the five most current generations
for number of ant generations g do
    for each ant a do
        //Let x_{g,a} be the ant score of ant a from generation g
    sum += x_{g,a}
    end for
end for
```

$$\mu = \frac{sum}{(\#\ of\ ant\ generations)(\#\ of\ ants)}$$

**//Compute and store mapped values into array**<br>
**for** number of ant generations g **do**<br>
    **for** each ant a **do**

$$array_{g,a} = \left[\frac{\lambda}{2\pi x_{g,a}{}^{3}}\right]^{\frac{1}{2}} exp\frac{-\lambda(x_{g,a}-\mu)^2}{2\mu^2 x_{g,a}}\quad for\ x > 0$$

    **end for**<br>
**end for**<br>
//**Normalize values in array**<br>
*normalArray = normalize(array)*<br>
**//Sum up all values**<br>
**for** number of ant generations g **do**<br>
    **for** each ant a **do**<br>
        $qSum \mathrel{+}= normalArray_{g,a}$<br>
    **end for**<br>
**end for**

$$q_0 = \frac{qSum}{(\#\ of\ ant\ generations)(\#\ of\ ants)}$$

**Figure 8. Pseudocode for the Dynamic Algorithm**

### 3.1.4 Distributed Ant Colony System Standard (DACSS)

The DACSS is the ACSS algorithm transitioned to solve in a distributed manner. In the distributed environment, each node only possesses local information (i.e. information about itself) and must act independently of the other nodes and the network as a whole. Each node makes its own edge connection choices using their potential outgoing edge list. Each node runs an ACS algorithm (DACSS or DACSE) to select the potential outgoing edges to be converted to regular outgoing edges. Each node performs its own local pheromone update based on the edges it selected. However, the global update rule is done after the final network topology is evaluated at the global level. This global evaluation verifies that the commodities are properly being routed through the network. The components of the DACSS algorithm are as follows:

- Problem representation is the entire network graph. Let $N$ denote the set of nodes, $K$ the number of commodities, $F$ the number of interface types, and G the number of groups. Further, let $e_{ijf}$ denote the edge connecting node $i$ to node $j$ by the interface type $f$. The search space is all of each node's set of potential outgoing edges, $e_{ijf} \in E_i$ for a given node $i$. The solution space $S_i$, consists of all potential outgoing edges, $e_{ijf}$ selected from $E_i$ as outgoing edges, $s_{ijf}$. The desirable solution $S$ is the one with the minimum total cost. The fixed edge cost is $c_{ijf}$ and the variable flow cost is $v_{ijf}^k$, which is the per unit cost of commodity $k$ on edge $e_{ijf}$. If $m^k = 1$, then commodity $k$ is dropped. The penalty function is 1000 for each dropped commodity multiplied by the commodities demand. The demand of a commodity is denoted $d_{ijf}^k$, the demand of commodity $k$ on $e_{ijf}$. Also, a group is defined as the set of all potential edges, $e_{ijf}$ from a specific node $i$ sharing the same interface type $f$. Also, a group is defined as a subset of $E$, $G \subset E$ and is the set of all potential edges, $e_{ijf}$ from a specific node $i$ sharing the same interface type $f$. Furthermore, only one potential edge $e_{ijf} \in G_{if}$ is able to be selected from within a group to be added to the solution set $S$.
- The heuristic calculation for DACSS is the same as ACSS:

$$\eta_{ijf} = \frac{1}{c_{ijf}} \qquad [49]$$

where $c_{ijf}$ is the fixed edge cost of using edge $e_{ijf}$ in the network.

- The evaluation criteria for the network topology solution are the same as ACSS and rely on actually routing all commodities through the network topology to compute the overall cost. Each node sends its edge selections forward to be evaluated at the global level. The specific network flow routing algorithm is the Edmonds Karp, best-first search greedy selection (Garner, 2007). This particular routing method has the overall best run time and performance when used with ACSS (details of this testing appear in Table 22 and Table 38). The cost of the network topology is a composite score based upon the fixed edge costs, the commodity flow cost and any penalties associated with dropped commodities (see ACSS Equation 34).
- Pheromone is initialized by depositing a small amount of pheromone, $\tau_0$ in every potential edge of the network graph. This initial amount is still calculated by computing the network cost using the Edmonds Karp, best-first search greedy selection. The initialization of $\tau_0$ still occurs at the global level when the algorithm begins (see ACSS Equation 35).
- The probabilistic transition rule is similar to the ACSS algorithm. Pheromone is modeled by means of a matrix $\tau$, where $\tau_{ijk}$ contains the level of pheromone deposited in the edge from node $i$ to node $j$ on interface $k$. An ant $a$ selects the next available edge to add to the node's outgoing edge list by:

$$p_{ijf}^{a} = \begin{cases} \dfrac{\left[\tau_{ijf}\right]^{\alpha}\left[\eta_{ijf}\right]^{\beta}}{\sum_{e_{ijf} \in E^{a}}\left[\tau_{e_{ijf}}\right]^{\alpha}\left[\eta_{e_{ijf}}\right]^{\beta}} & \text{if } e_{ijf} \in E_i \\ 0 & \text{otherwise} \end{cases} \quad [50]$$

where $\eta_{ijk}$ represents heuristic information about the problem; $E_i$ is the set of all available potential edges at node $i$; $\alpha$ and $\beta$ are two parameters that determine the relative importance of the pheromone with respect to the heuristic information.

- The local update (ACSS Equation 37) is the same as ACSS, but is accomplished after each node makes an edge selection.
- The global update (ACSS Equation 38) remains the same as ACSS because the evaluation occurs at the global level. After a "best solution" is found, the global update information is provided to the nodes to actually update their pheromone matrix.

The pseudocode for the DACSS algorithm is shown in Figure 9.

**// Initialization**<br>
*route commodities using EdmondKarp1, greedy selection*<br>
$\tau_0 = \frac{1}{c_{network}}$<br>
**for** each node $n_i \in N$ **do**<br>
    create a thread for $n_i \in N$<br>
    **for** each edge $e_{ijf} \in E_i$ **do**<br>
        $\tau_{ijf} = \tau_0$<br>
    **end for**<br>
    //**Main loop**<br>
    **for** total number of iterations **do**<br>
        **for** each ant *a* **do**<br>
            **for** each potential edge $e_{ijf} \in E_i$ **do**<br>
                $e_{ijf} = visited(FALSE)$<br>
            **end for**<br>
            **while** $\exists\, e_{ijf} \in E_i = visited(FALSE)$ **do**<br>
                *q = rand[0,1]*<br>
                **// Get transition probabilities**<br>
                **for** each potential edge $e_{ijf} \in E_i$ **do**

$$p_{ijf}^{a} = \begin{cases} \dfrac{\left[\tau_{ijf}\right]^{\alpha}\left[\eta_{ijf}\right]^{\beta}}{\sum_{e_{ijf} \in E^{a}}\left[\tau_{e_{ijf}}\right]^{\alpha}\left[\eta_{e_{ijf}}\right]^{\beta}} & \text{if } ijf \in E_i \\ 0 & \text{otherwise} \end{cases}$$

**end for**

**//Make edge selection**

**if** *($q \leq q_0$)* **then**

$s_{ijf}$ = min( $p_{ijf}^{a}, e_{ijf} \in E_i$ )

**else**

*randomly select edge* $s_{ijf}, e_{ijf} \in E_i$

**end if**

**//All edges in selected group to visited is true**

$\forall\, e_{ijf} \in E_i^{g} = visited(TRUE)$

**// Update local pheromone**

$\tau_{ijf} = 1-\xi \;\; \tau_{ijf} + \xi\tau_0$

**end while**

**return** $S_i$ to Network Event Manager

**//Evaluation done at global/network level**

*evaluate solution using a network flow algorithm*

**if** (*currentScore* < *bestScore*) **then**

*bestScore* = *currentScore*

*bestNetwork* = *currentNetwork*

**end if**

**end for**

**// Update global pheromone for best ant solution**

**for** each $s_{ijf} \in S$

$\tau_{ijf} = 1-\rho \;\; \tau_{ijf} + \rho\Delta\tau_{ijf}$

**end for**

**end for**

**end for**

**Figure 9. Pseudocode for the DACSS Algorithm**

Although each node is independently selecting the edges to be used for the network topology, the selected edges from the node level are passed back to the network object. The network object then consolidates all of the nodes' edges to construct the network topology and perform the routing to effectively evaluate the solution. The

primary difference between the ACSS and DACSS algorithms is how edge selection occurs. With ACSS, the ACS algorithm is applied to the network, evaluating all the potential edges with global knowledge of the entire network. However, each node has no knowledge of which edges are being selected until the final network topology is returned. Figure 9 illustrates this approach. With the distributed approach, each node runs its own ACS algorithm to select the edges to be added to the network. The nodes act independently with no global knowledge of what the other nodes select, shown in Figure 10.

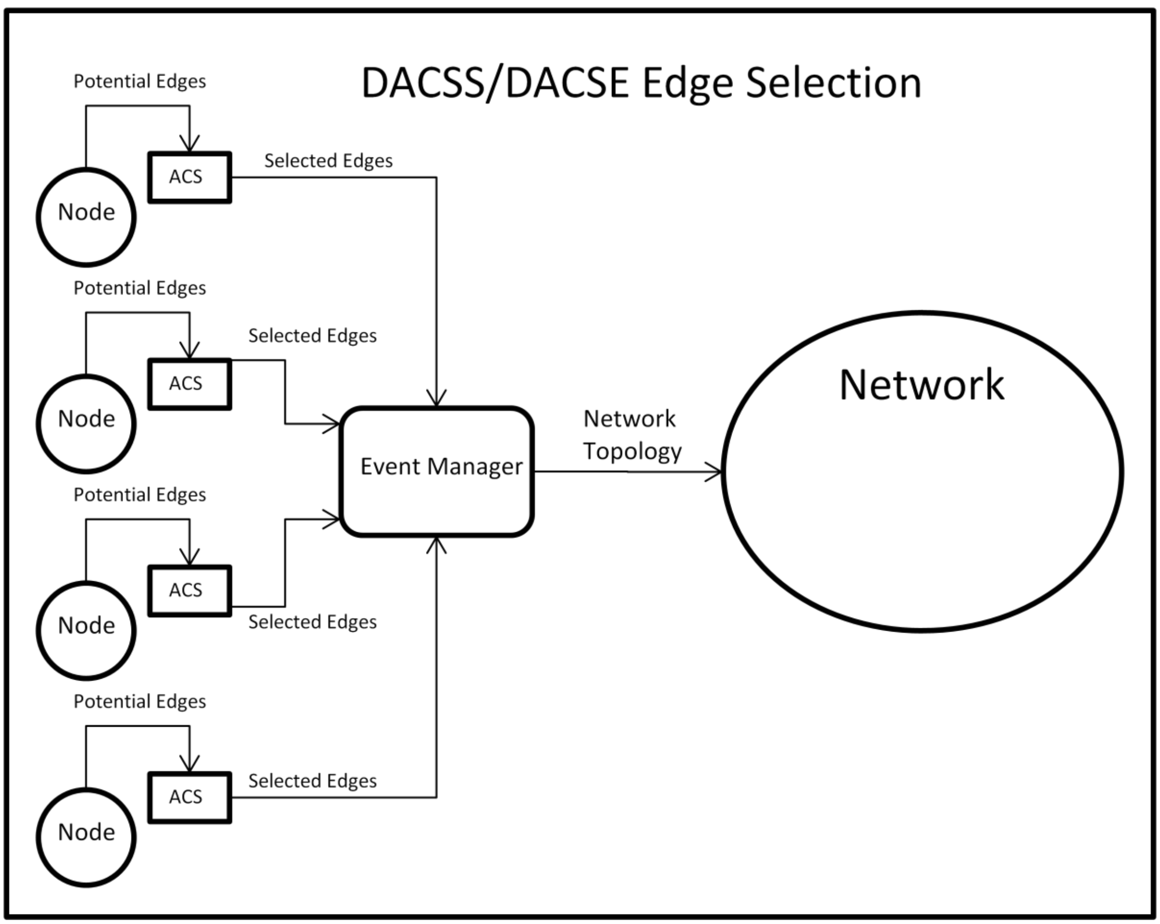


**Figure 10. DACSS/DACSE Edge Selection**

The algorithm continues to select until there are no available edges remaining. After evaluation of the solution, if an edge is selected, but has no flow it is removed from the network topology.

### 3.1.5 Distributed Ant Colony System Estimation (DACSE)

The DACSE is comparable to the ACSE approach discussed in 3.1.2. The results of the Phase 1 ACSE testing identified which heuristics produced solutions with the minimum total cost. These heuristics were then chosen to be used for the DACSE testing for the 10 and 15-Node experiments. Similar to the ACSS/DACSS comparison, the primary difference between the DACSE and ACSE is how the edges are selected (Figures 9 and 10 illustrate this). The components of the DACSE algorithm are similar to the ACSE algorithm but with the following modifications:

- Problem representation is the same as DACSS.
- The following DACSE heuristics and heuristic combinations are used:

1. The fixed edge heuristic is:

$$\eta_{ijf} = \frac{1}{c_{ijf}} \qquad [51]$$

where $c_{ijf}$ is the fixed edge cost of edge $e_{ijf}$.

2. Weighted sum of fixed cost and edge capacity heuristic:

$$\eta_{ijf} = (p_1)\frac{1}{c_{ijf}} + (p_2)e^{c}_{ijf} \qquad [52]$$

where $e^{c}_{ijf}$ is the capacity of edge $e_{ijf}$. The values for $p_1$ and $p_2$ are listed in Table 2. This DACSS equation corresponds to Test Case 14 in Table 2.

- The evaluation criteria for the proposed network topology use a heuristic value of the fixed edge cost, $c_{ijf}$ (equation 51) for both the 10-node and 15-node tests to estimate the routing of all commodities through the network topology. Each node sends its edge selections forward to be evaluated at the global level. However, once a final network topology is selected, the final network score is constructed using the Edmonds Karp, greedy selection routing algorithm to include the fixed edge cost, the commodity flow cost and any penalties associated with dropped commodities (same as ACSS, see equation 34).
- Pheromone is initialized the same as ACSE (Equation 47). This is accomplished at the global level to ensure the initial value for $\tau_0$ is the same for all nodes and is more reflective of an actual network evaluation score.
- The probabilistic transition rule is also similar to DACSS (Equation 50). However, the DACSE 10-node tests uses heuristic 2 (Equation 52) and DACSE 15-node tests uses heuristic 1 (Equation 51).
- The local update (ACSS Equation 37) is the same, but is accomplished after each node makes an edge selection.
- The global update (ACSS Equation 38) remains the same as ACSS/ACSE because the evaluation occurs at the global level. After a "best solution" is found, the global update information is provided to the nodes to actually update their pheromone matrix.

The pseudocode for the DACSE algorithm is presented in Figure 11Figure 11.

**// Initialization**
**for** each node $n_i \in N$ **do**
    **for** each potential edge $e_{ijf}$ in $n_i$ **do**
    insert $e_{ijf}$ into potential edge list $E_i$
$\tau_0 = \dfrac{1}{\left(\sum_{i=1}^{n}\sum_{f=1}^{3} min(c_{ijk})\forall j\right)}$
**for** each node $n_i \in N$ **do**
    create a thread for $n_i \in N$
    **for** each edge $e_{ijf} \in E_i$ **do**
        $\tau_{ijf} = \tau_0$
    **end for**
    //**Main loop**
    **for** total number of iterations **do**
        **for** each ant $a$ **do**
            **for** each potential edge $e_{ijf} \in E_i$ **do**
                $e_{ijf} = visited(FALSE)$
            **end for**
            **while** $\exists\, e_{ijf} \in E_i = visited(FALSE)$ **do**
                $q = rand[0,1]$
                **// Get transition probabilities**
                **for** each potential edge $e_{ijf} \in E_i$ **do**

$$p_{ijf}^{a} = \begin{cases} \dfrac{\left[\tau_{ijf}\right]^{\alpha}\left[\eta_{ijf}\right]^{\beta}}{\sum_{e_{ijf}\in E^{a}}\left[\tau_{e_{ijf}}\right]^{\alpha}\left[\eta_{e_{ijf}}\right]^{\beta}} & \text{if } ijf \in E_i \\ 0 & \text{otherwise} \end{cases}$$

                **end for**
                **//Make edge selection**
                **if** $(q \le q_0)$ **then**
                    $s_{ijf}$ = $\min(p_{ijf}^{a}, e_{ijf} \in E_i)$
                **else**
                    *randomly select edge* $s_{ijf}, e_{ijf} \in E_i$
                **end if**
                **//All edges in selected group to visited is true**
                $\forall\, e_{ijf} \in E_i^{g} = visited(TRUE)$
                **// Update local pheromone**
                $\tau_{ijf} = 1 - \xi \;\; \tau_{ijf} + \xi\tau_0$
            **end while**

*evaluate solution using a heuristic method*
**if** (*currentScore* < *bestScore*) **then**
*bestScore* = *currentScore*
*bestNetwork* = *currentNetwork*
**end if**
**end for**
**// Update global pheromone for best ant solution**
**for** each $s_{ijf} \in S$
$\tau_{ijf} = 1-\rho \;\; \tau_{ijf} + \rho\Delta\tau_{ijf}$
**end for**
**end for**
**end for**

**Figure 11. Pseudocode for the DACSE Algorithm**

### 3.2 Test Environment

Experiments were completed on a desktop personal computer, running CentOS release 4.6 x86_64. The processor is a 2xDual-Core AMD Opteron (tm) processor 2218 (1.8 GHz) with 4 GB of memory. All code was written in C/C++ and compiled and tested using the GNU Compiler Collection, version 4.0.0.

### 3.3 Parameter Settings

Sensitivity analysis was not performed on the parameters identified in Table 4. However, all parameter settings that are common to a majority of the ACO algorithms are initialized using values found in previous experimental studies (Dorigo & Stutzle, 2004):

**Table 4. Parameter Settings**

| Parameter Settings | | |
|---|---|---|
| **Parameter** | **Value** | **Description** |
| $\eta_{ijf}$ | Varies based on heuristic used | Desirability of edge $e_{ijf}$ |
| $\alpha$ | 1 | Influence parameter of $\tau$ |
| $\beta$ | 2 | Influence parameter of $\eta$ |
| $\rho$ | 0.1 | Rate of global evaporation |
| $\xi$ | 0.1 | Rate of local evaporation |
| $\tau_0$ | See Equations 35 and 47 | Initial pheromone value |
| $q_0$ | 0.9 (Baseline and Restart only) | Exploration/Exploitation parameter |

### 3.4 Summary

This chapter presents several new methods for solving the MCNDP with both weak and strong dynamic network change. In Phase 1, the first two methods developed, ACSS and ACSE present two ways to apply ACS to the MCNDP problem. Phase 2 introduces dynamics to the MCNDP and the alterations to the ACSS and ACSE algorithms to better handle strong dynamic changes. This is followed by a discussion of distributed methods for solving the problem. In the target environment of highly mobile devices it doesn't make sense for a central solver to receive all of the information and issue a solution. In addition, all the information would have to be made available to the central location. Instead, each node makes its own decisions with the distributed results being of the same quality of solution.

The following chapter presents the analysis and results of the centralized solvers (ACSS and ACSE) compared to the MILP (Erwin, 2006) and Network Flow Routing algorithms (Garner, 2007). The chapter continues with comparison results of the three dynamic categories used to solve the DMCNDP. For the distributed solvers, comparisons

test results between the centralized approach and the distributed approach to the DMCNDP are discussed with respect to network topology cost and computation time.

# IV. Analysis and Results

The goal of this research is to solve the Multi-commodity Capacitated Network Design Problem (MCNDP) and the Dynamic MCNDP (DMCDNP) using Ant Colony Optimization techniques in a highly dynamic environment using both a centralized and distributed solver.  The first experiments consist of comparing the Ant Colony System Standard (ACSS) and Ant Colony System Estimation (ACSE) algorithms with a Mixed Integer Linear Program (MILP) (Erwin, 2006) and Network Flow solvers (Garner, 2007). Initial application of the Ant Colony System (ACS) algorithms produce near optimal results when solving the MCNDP. All four of the ACSS algorithms are able to find a "best" solution with no dropped commodities for both a 10-node and 15-node network. The ACSE approach demonstrated the ability to find low cost solutions in a fraction of the time. However, the heuristics are not consistent across all routing algorithms, but do approximate the ACSS solutions in several test cases.

The second phase of testing focuses on the DMCNDP and specifically the performance of ACSS and ACSE while network changes ranges from mild to drastic. Testing of these algorithms with the DMCNDP included transforming the network environment from static to highly dynamic. The three different solution methods presented in Chapter 3, demonstrate that ACSS and ACSE are efficient and adapt well to a dynamically changing network environment. Three categories of both the ACSS and ACSE algorithms are analyzed to determine how they adapt to a highly dynamic network environment. Convergence charts illustrate how the algorithms respond throughout all four intervals of change.

The last testing phase evaluates the distributed approaches, DACSS and DACSE, to solve the DMCNDP. The distributed results, with respect to solution cost and computation time, are compared with the centralized results to the DMCNDP. The statistical tool applied for analysis of the centralized and distributed solutions is the two tailed student *t*-test. In addition, the computation times of the centralized and distributed solvers are compared to determine the efficiency of the distributed approach.

## 4.1 Metrics

Four metrics were collected for the Phase 1 testing, comparing the ACSS and ACSE algorithms with the MILP and Network Flow solvers. The metrics are, the average number of hops generated per solution, the number of dropped commodities, total cost of the network, and network diameter. However, for all of the Phase 2 and Phase 3 tests (dynamic and distributed) only Metric 3 (Total Cost) is evaluated.

***Metric 1: Number of Hops*** - This metric measures the average number of hops per commodity. This particular metric is associated with delay, a common metric associated with routing problems.

***Metric 2: Dropped Commodities*** - The average number of dropped commodities indicates how effective the network is performing. This is equivalent to network throughput, the amount of data that is in flow at any one time. In addition, dropped commodities have a penalty associated with them which raises the overall cost of the selected network topology

***Metric 3: Total Cost*** - The cost for constructing the network is the sum of the cost of constructing the links in the network (fixed costs), the cost to route the

solved commodities across each of the links (variable costs), and any penalties associated with dropped commodities.

***Metric 4: Network Diameter*** - The final metric is related to the number of hops per commodity. This provides a general indication of how efficient a network is constructed. As the number of nodes increases, the network diameter is expected to remain relatively small.

## 4.2 Centralized Test Results

The first phase of testing evaluates the ACSS and ACSE algorithms on a static MCNDP. Specifically, the testing compares the ACSS and ACSE solution for 10-node and 15-node networks with the results produced by a MILP (Erwin, 2006) and network flow routing algorithms (Garner, 2007). To ensure meaningful results from testing the ACSS and ACSE solutions, the same test input files (10 and 15-Node networks) used by (Erwin, 2006) and (Garner, 2007) are evaluated. Also, the same metrics (Section 4.1) are captured for comparison. For the ACSS implementation, only the four Greedy algorithms are run due to time and memory limitations. However, for the ACSE approach, 14 test experiments, comparing different heuristic weights, are run in order to identify the best weights and tradeoffs between heuristic elements. For each experiment, 30 trials are run and the average and standard deviations computed over four metrics. Data for the MILP comparison is obtained from (Erwin, 2006). However, the network flow algorithms were retested to ensure the most accurate and reliable data is available. The metrics identified in Section 4.1 are captured and compared with two previous solution techniques (Erwin,

2006) and (Garner, 2007). Appendix A contains all Phase 1, 10 Node test results and Appendix B contains all Phase 1, 15 Node test results.

### 4.2.1 Ant Colony System Standard (ACSS)

The ACSS approach is tested using both a 10-node and 15-node multi-commodity capacitated network. The ACSS algorithm is compared against the four greedy Maximum Flow approaches discussed in (Garner, 2007) and the MILP solution (Erwin, 2006).

More testing details discovered that the MILP results are invalid. During the process of verifying test results, it was discovered that the MILP does not return complete network topologies and failed to accurately route based on valid and feasible solution topologies from other solvers. However, the results of the MILP solver are included in this paper for completeness and are not indicative of the optimal solution that they should return. Table 5 shows a comparison of the MILP, Maximum Flows (Garner, 2007), and ACSS implementations with respect to the four metrics previously identified in section 3.3 for a 10-node network and Table 6 shows comparative results for a 15-node network. For the 10-node network, ACSS produced better solutions compared to all previous approaches except for MILP methods (Barrier, Dual, and Primal) (Erwin, 2006). For the 15-node network, ACSS outperformed all previous approaches. Most significant is the fact that ACSS did not drop a single commodity providing an extremely low cost and reliable network topology.

**Table 5. 10 Nodes - Averages**

| Approach | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| LP Barrier* | 161.30 | 662.94 | N/A | 824.24 | 56.33 | 2.07 | 5.60 | 0.00 | 20.89 |
| LP Dual* | 161.30 | 662.94 | N/A | 824.24 | 56.32 | 2.07 | 5.60 | 0.00 | 17.96 |
| LP Primal* | 161.30 | 662.94 | N/A | 824.24 | 56.32 | 2.07 | 5.60 | 0.00 | 19.31 |
| Combo* | 139.60 | 721.42 | N/A | 861.02 | 72.72 | 2.12 | 5.50 | 0.00 | 1.71 |
| Heuristic 1* | 144.50 | 742.83 | N/A | 887.33 | 61.08 | 2.20 | 6.00 | 0.00 | 0.48 |
| Heuristic 2* | 143.90 | 725.10 | N/A | 869.00 | 86.55 | 2.22 | 5.60 | 0.00 | 0.54 |
| Knapsack EK1** | 123.27 | 1126.07 | 2400.00 | 3649.34 | 4246.44 | 3.50 | 6.80 | 0.73 | 4.12 |
| Knapsack EK2** | 139.00 | 1107.98 | 3433.33 | 4680.32 | 5630.41 | 4.01 | 8.13 | 0.97 | 5.72 |
| Knapsack PFP1** | 176.97 | 906.42 | 0.00 | 1083.39 | 60.48 | 2.99 | 5.40 | 0.00 | 7.65 |
| KnapsackPFP2** | 190.03 | 881.22 | 500.00 | 1571.25 | 1527.75 | 2.93 | 5.60 | 0.10 | 12.71 |
| Greedy EK1** | 127.83 | 1125.89 | 2566.67 | 3820.39 | 4444.37 | 3.49 | 6.90 | 0.77 | 1.41 |
| Greedy EK2** | 139.63 | 1159.78 | 2633.33 | 3932.74 | 3891.77 | 4.07 | 7.97 | 0.80 | 1.83 |
| Greedy PFP1** | 169.50 | 928.70 | 133.33 | 1231.54 | 742.02 | 3.04 | 5.63 | 0.03 | 2.37 |
| Greedy PFP2** | 186.33 | 871.38 | 1933.33 | 2991.04 | 5983.86 | 2.93 | 5.33 | 0.43 | 4.28 |
| Greedy EK1 ACSS | 188.67 | 701.37 | 0.00 | 890.03 | 10.35 | 2.60 | 4.00 | 0.00 | 1547.16 |
| Greedy EK2 ACSS | 187.86 | 723.98 | 0.00 | 911.84 | 17.95 | 3.77 | 7.41 | 0.00 | 1926.58 |
| Greedy PFP1 ACSS | 176.73 | 693.33 | 0.00 | 870.06 | 8.11 | 2.63 | 4.00 | 0.00 | 2151.04 |
| Greedy PFP2 ACSS | 178.63 | 692.25 | 0.00 | 870.88 | 7.29 | 2.62 | 4.00 | 0.00 | 2140.26 |

* - MILP total cost does NOT include the penalty cost. Data obtained from (Erwin, 2006).
** - Network Flow algorithms (Garner, 2007) were re-tested to include the dropped penalty cost.

**Table 6. 15 Nodes - Averages**

| Approach | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| LP Barrier* | 302.20 | 1579.01 | N/A | 1881.21 | 120.00 | 2.38 | 7.30 | 5.80 | 707.27 |
| LP Dual* | 302.00 | 1616.31 | N/A | 1918.31 | 161.64 | 2.41 | 7.50 | 5.70 | 762.57 |
| LP Primal* | 291.80 | 1638.60 | N/A | 1930.40 | 175.92 | 2.42 | 7.10 | 5.80 | 867.87 |
| Combo* | 264.50 | 1602.55 | N/A | 1867.05 | 117.97 | 2.40 | 7.40 | 8.60 | 142.94 |
| Heuristic 1* | 245.30 | 1703.92 | N/A | 1949.22 | 176.04 | 2.55 | 8.20 | 16.80 | 3.55 |
| Heuristic 2* | 250.30 | 1640.68 | N/A | 1890.98 | 151,38 | 2.51 | 7.80 | 19.00 | 3.64 |
| Knapsack EK1** | 264.20 | 3195.14 | 6666.67 | 10126.01 | 8079.66 | 4.01 | 9.50 | 1.70 | 45.03 |
| Knapsack EK2** | 281.43 | 3033.35 | 13533.33 | 16848.12 | 11913.31 | 4.73 | 10.70 | 3.77 | 64.34 |
| Knapsack PFP1** | 321.93 | 2400.28 | 3266.67 | 5988.88 | 7298.50 | 3.28 | 6.80 | 0.83 | 77.38 |
| KnapsackPFP2** | 334.83 | 2240.27 | 21066.67 | 23641.75 | 22157.38 | 3.18 | 6.77 | 6.33 | 197.77 |
| Greedy EK1** | 252.33 | 3208.69 | 5800.00 | 9261.03 | 9095.95 | 4.06 | 9.53 | 1.43 | 13.21 |
| Greedy EK2** | 270.77 | 3016.27 | 16933.33 | 20220.38 | 11768.29 | 4.72 | 10.80 | 4.70 | 19.20 |
| Greedy PFP1** | 318.83 | 2490.44 | 2766.67 | 5575.94 | 6628.40 | 3.37 | 7.33 | 0.80 | 23.43 |
| Greedy PFP2** | 327.27 | 2256.37 | 16866.67 | 19450.31 | 15486.60 | 3.17 | 6.57 | 4.93 | 49.24 |
| Greedy EK1 ACSS | 282.57 | 1784.18 | 0.00 | 2066.75 | 22.90 | 2.79 | 4.13 | 0.00 | 8380.81 |
| Greedy EK2 ACSS | 285.00 | 1787.72 | 0.00 | 2072.72 | 0.00 | 4.30 | 9.78 | 0.00 | 11439.74 |
| Greedy PFP1 ACSS | 277.97 | 1843.54 | 0.00 | 2121.50 | 29.41 | 2.83 | 4.57 | 0.00 | 12625.34 |
| Greedy PFP2 ACSS | 277.71 | 1867.65 | 0.00 | 2145.36 | 39.54 | 2.84 | 4.71 | 0.00 | 12386.57 |

* - MILP total cost does NOT include the penalty cost. Data obtained from (Erwin, 2006).
** - Network Flow algorithms (Garner, 2007) were re-tested to include the dropped penalty cost.

As previously noted, the ACSS implementations on both the 10-node and 15-node networks construct network topologies with zero dropped commodities and significantly reduce the overall total cost. The runtime of the ACSS algorithm is significantly higher than (Erwin, 2006) and (Garner, 2007) solutions due to the fact that the ACSS algorithm calculates an actual route for each ant solution generated to evaluate the network. The total number of routes is the product of the total number of ants (which is set to equal the number of nodes) and the number of iterations. The number of iterations was set at 35

(due to memory constraints at the time of testing), but increased to 100 iterations for both the dynamic and distributed research.

For the 10-node problem, the ACSS approach has overall cost solutions that are comparable, but slightly higher than the MILP methods (Erwin, 2006). However, the overall cost solutions compared to the best Network Flow solution (Garner, 2007) for the 10-node problem are almost twenty percent less. In addition, the number of hops and network diameter are slightly higher compared to (Erwin, 2006) and slightly less when compared to (Garner, 2007).

For the 15-node problem, the ACSS approach outperformed both (Erwin, 2006) and (Garner, 2007) approaches. The ACSS overall cost solution is significantly lower, almost one-third of the cost when compared to the Network Flow best solution. The MILP solution did not include the penalty cost for dropped commodities in its solutions, but all tests had solutions with dropped commodities. The penalty for each dropped commodity is $p^k = 1000(m^k)(d_{ijf}^k)$ where 1000.0 is the base penalty multiplied by a dropped commodity $m^k$ which is 0 or 1, multiplied by the demand of commodity $k$, from node $i$ to node $j$ using interface $f$. Therefore, the MILP solutions are at least as expensive if not more as the ACSS solutions. The network diameter is lower for all of the ACSS solutions except the Greedy Edmonds Karp 2 algorithm. The number of hops is comparable with (Erwin, 2006) and slightly less than (Garner, 2007).

### 4.2.2 Ant Colony System Estimation (ACSE)

The ACSE algorithm is tested using a 10-node and 15-node multi-commodity capacitated network. The ACSE algorithm is compared with the four knapsack and four

greedy Maximum Flow approaches presented in (Garner, 2007) and the Mixed Integer Linear Program (MILP) methods (Erwin, 2006). There are 14 ACSE test cases for both the 10-node and 15-node networks as outlined in Table 2. The fixed edge cost heuristic was the first choice to use with both the probability transition rule and the solution evaluation function. The fixed edge heuristic component is the cost of an edge from node $i$ to node $j$ using interface $f$ (Equation 39). A second heuristic is constructed using the sum of the fixed edge cost and the average commodity cost per edge (Equation 42). A third heuristic is computed using the fixed edge cost and the capacity of an edge from node $i$ to node $j$ using interface $f$ (Equation 43). The fourth heuristic utilizes fixed edge cost and the value of an edge with respect to the commodities that potentially flow on an edge from node $i$ to node $j$ using interface $j$ (Equation 46).

Test 1 (Table 2) uses the sum of the fixed edge cost and the variable edge cost as the heuristic for both the transition probability rule and the evaluation criteria. Test 2 uses only the fixed edge cost for both. Tests 3 – 5 use a weighted combination of the fixed edge cost and the edge capacity for both the transition probability rule and the evaluation criteria. However, for Tests 6 – 9, only the fixed edge cost is used for the evaluation criteria. Similarly, Tests 9 – 11 use the same weighting criteria for both the transition probability rule and the evaluation criteria, but are based on the fixed edge cost and the edge value (vice previously using the edge capacity). The last three (Tests 12 – 14), use only the fixed edge cost for the evaluation criteria. Overall, there is improvement when compared to Network Flow (Garner, 2007) results. With the 10-node ACSE tests, results show improvement, but not near the quality of the ACSS approach. With the 15-node tests, there is again improvement when compared to the Network Flow (Garner, 2007)

results. More importantly, heuristic values were generated for the ACSE algorithms that produce solutions with zero dropped commodities, generating cost solutions similar to ACSS. More importantly, the solutions are being constructed faster, significantly reducing the overall computation time. Table 7 identifies each routing algorithm and the ACSE heuristic that produces the best result for that particular routing algorithm for a 10-node network.

**Table 7. Best ACSE Solutions (10 Nodes).**

| ACSE Method | Test Case | Num of Hops | Drop Comm | Net Diam | Run Time | Total Cost | SD | ACSS Cost | Iteration Found | SD |
|---|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 10 | 2.66 | 0.00 | 4.00 | 26.16 | 927.23 | 8.34 | N/A | 15.90 | 18.44 |
| Knapsack EK2 | 4 | 3.84 | 0.00 | 8.00 | 25.01 | 1027.41 | 20.83 | N/A | 11.17 | 9.13 |
| Knapsack PFP1 | 5 | 2.68 | 0.00 | 4.00 | 27.58 | 925.68 | 4.55 | N/A | 17.23 | 24.62 |
| Knapsack PFP2 | 2 | 2.68 | 0.00 | 4.00 | 28.14 | 927.93 | 11.10 | N/A | 3.77 | 4.72 |
| Greedy EK1 | 5 | 2.67 | 0.00 | 4.00 | 21.90 | 931.74 | 7.46 | 890.03 | 14.33 | 14.17 |
| Greedy EK2 | 14 | 3.96 | 0.00 | 7.53 | 25.47 | 1021.45 | 5.28 | 911.84 | 16.20 | 25.57 |
| Greedy PFP1 | 14 | 2.69 | 0.00 | 4.00 | 25.00 | 920.61 | 3.63 | 870.06 | 15.43 | 18.27 |
| Greedy PFP2 | 14 | 2.69 | 0.00 | 4.00 | 25.56 | 921.09 | 3.53 | 870.88 | 11.77 | 21.36 |

Table 8 shows the identical information for a 15-node network. Although all eight Network Flow routing algorithms were evaluated, future testing only uses the Edmond-Karp Greedy 1 algorithm as it provides quick and reasonably favorable results. For the 10-node network results, ACSE identifies network solutions with zero dropped commodities and an overall network cost very comparable to ACSS. Test results with the 15-node network were optimistic. With the Greedy Pre Flow Push 1, ACSE has less than 1 dropped commodity and again the solution was comparable with ACSS. The Knapsack

Pre Flow Push 1 had zero dropped commodities. In four out of the remaining six tests, the average number of dropped commodities was less than 2, very remarkable for estimating the total cost vice actually routing the commodities.

**Table 8. Best ACSE Solutions (15 Nodes).**

| ACSE Method | Test Case | Num of Hops | Drop Comm | Net Diam | Run Time | Total Cost | SD | ACSS Cost | Iteration Found | SD |
|---|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 13 | 2.86 | 1.70 | 5.00 | 208.49 | 9594.15 | 5481.43 | N/A | 49.57 | 29.47 |
| Knapsack EK2 | 13 | 4.50 | 4.57 | 10.47 | 245.97 | 18224.48 | 8396.52 | N/A | 46.90 | 27.97 |
| Knapsack PFP1 | 3 | 2.90 | 0.00 | 4.57 | 203.92 | 2377.14 | 16.38 | N/A | 38.06 | 29.48 |
| Knapsack PFP2 | 4 | 2.90 | 1.10 | 5.03 | 216.87 | 5571.89 | 861.53 | N/A | 47.97 | 22.87 |
| Greedy EK1 | 2 | 2.86 | 1.60 | 4.90 | 162.36 | 8440.15 | 5218.43 | 2066.75 | 79.20 | 22.50 |
| Greedy EK2 | 12 | 4.53 | 4.57 | 10.27 | 170.51 | 16349.61 | 6605.51 | 2072.72 | 43.80 | 26.86 |
| Greedy PFP1 | 12 | 2.91 | 0.20 | 5.00 | 176.11 | 3081.69 | 1619.31 | 2121.50 | 50.50 | 24.72 |
| Greedy PFP2 | 6 | 2.90 | 1.47 | 5.00 | 167.23 | 8205.74 | 2586.56 | 2145.36 | 50.67 | 27.74 |

### 4.2.3 ACSS/ACSE Improvements

Using ACSS to construct the network topology first and then using a network flow routing algorithm significantly reduces the overall cost. For the 10-node MCNDP, the average greedy network flow cost using ACSS was less than 30 percent of the total cost using just the greedy network flow methods (Garner, 2007). The 15-Node MCNDP was even more dramatic. The average greedy network flow cost using ACSS was only 15 percent of the total cost when compared with using the greedy methods alone to solve the 15-Node MCNDP. However, the run times of the ACSS approaches were significantly higher than any of the network flow methods.

The ACSE approach to build the network topology first and then use a network flow routing algorithm also lowers the total cost. The total cost to construct a 10-node MCNDP solution using ACSE was 32 percent of the total cost compared with all of the network flow solvers (Garner, 2007). ACSE for the 15-node MCDNP produced solutions at 65 percent of the cost compared to just using the network flow algorithms to route and develop the network topology. The computation time of ACSE compared to ACSS was less than 2 percent for both 10 and 15-Node MCNDP solutions.

### 4.3 Phase 2 Testing Results

The algorithms introduced in Phase 1 testing ACSS and ACSE are utilized while testing in the dynamic network environment. Testing is performed using both 10-node and 15-node networks. However, based on the Phase 1 test results, the number of routing algorithms used is reduced from eight to one. The Greedy, Edmonds-Karp 1 routing approach was selected based on producing the overall best solutions using ACSS during Phase 1 testing with respect to cost and the fastest solution times.

In order to create a dynamic environment, ten randomly generated test files were constructed. Each file was then modified with a designated percentage of change ranging from 0 percent to 50 percent in increments of 10. For a given percentage, the original file was modified three times. Each test begins execution with the same original set of test files. However, after each interval (25 iterations) a modified version of the previous test file is introduced to the testing environment. Figure 12 illustrates this process.

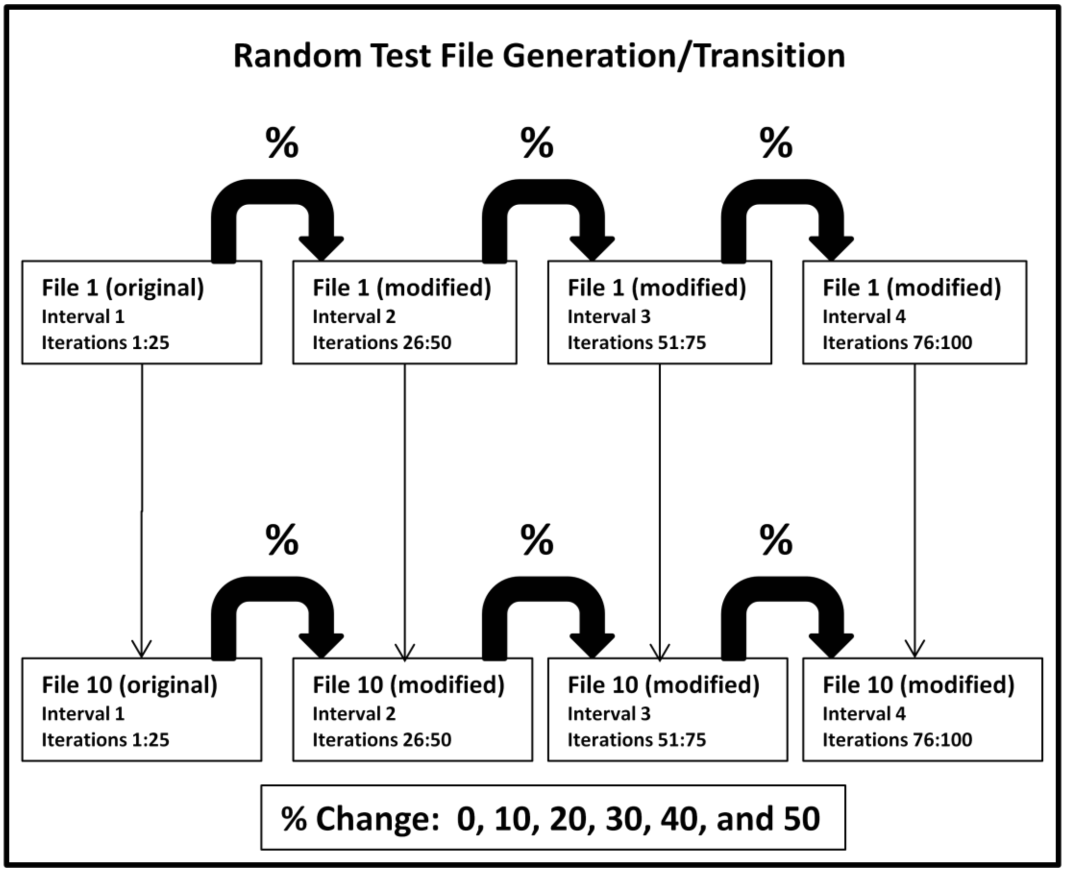


**Figure 12. Random Test File Generation/Transition Process**

The designated change percentage determines the amount of change to a file and the amount of change to the individual network parameters. Figure 13 shows the levels of change from weak to strong dynamics.

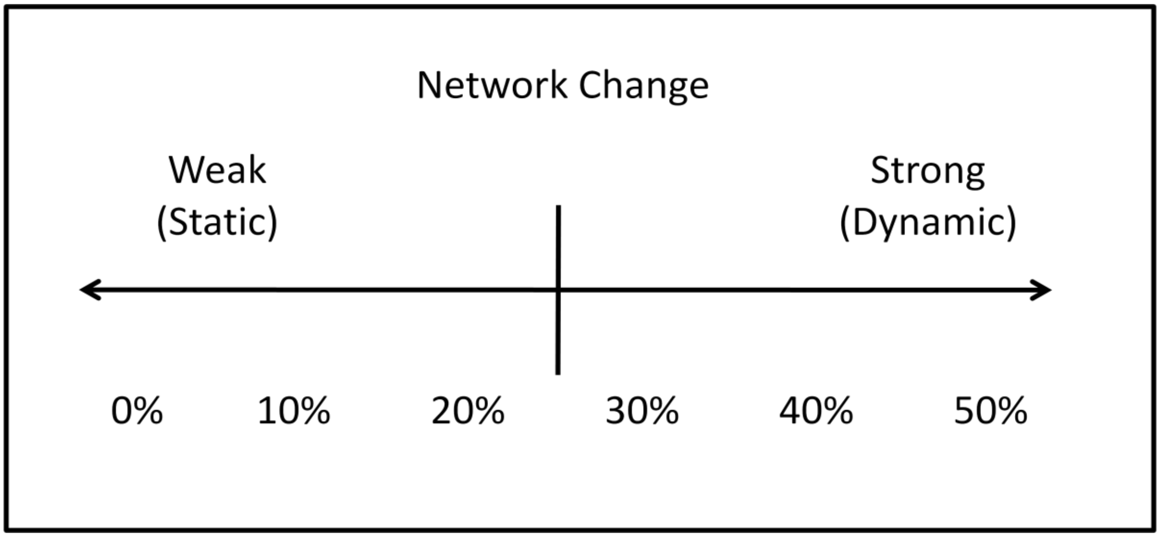


**Figure 13. Network Change**

In order to simulate a dynamically changing network environment, several network parameters are modified during algorithm execution. A total of six network parameters are modified during program execution. The specific values of each parameter being changed are randomly assigned. The six parameters are:

- Fixed edge cost
- Variable flow cost
- Demand of the commodity
- Edge capacity (bandwidth)
- Number and type of interfaces available
- Connectivity amongst nodes

The ACSS and ACSE algorithms, which were modified for the dynamic network environment, adapt to a variety of changes introduced to the network while a network topology solution is being constructed. The response to all of these dynamic variables is performed quickly and yet still provides an acceptable network topology solution. The ACO algorithms are designed to run 100 ant generations. However, after 25 iterations (an

interval) a new, modified test file is introduced which changes the network parameters and therefore creates the dynamic network environment. At each identified interval, network topology solutions are monitored to determine how the algorithms are adapting while the network experiences weak to strong dynamic change. A solution is computed and then compared to an optimal solution created by a MILP solver. Also, the convergence of the ACSS algorithms is evaluated to see how quickly the methods are finding their best solutions, especially when exposed to a network change. For each algorithm tested, there are 30 trials conducted for each of the ten randomly generated files with one exception. Testing for the 15-node network using ACSS and DACSS only has 10 trials.

Three new dynamic categories are introduced during this phase to compare how the ACSS and ACSE approaches handle strong dynamics within the MCNDP domain. These are the Baseline, Restart, and Dynamic category for both the ACSS and ACSE algorithms. For each category, a network score is obtained at the end of each interval to determine the ability of the algorithm to respond to change within the network. This section presents their results. In addition, several graphs illustrate each algorithm's convergence towards a solution. Appendix C includes all DMCNDP ACSS test results and Appendix D contains the DMCNDP ACSE test results.

### 4.3.1 Dynamic Results - Ant Colony System Standard (ACSS)

Of the three dynamic categories, the ACSS Dynamic (ACSS-D) had the lowest overall mean cost for nine of the ten test files. For a given file, there are six change percentages (0 percent, 10 percent, 20, percent, 30 percent, 40 percent, and 50 percent).

For each change percentage, there are four solution scores (1 solution score every 25 iterations or 1 per interval). Therefore, each algorithm has a total of 24 interval solution scores per file. These interval solution scores are the average of the 30 test iterations. However, for the ACSS-D and DACSS-D 15-node tests, there were only 10 test iterations completed. Regardless, an algorithm's overall mean solution cost for a file is computed by taking the average of the 24 mean interval solution scores. The data was then standardized and Figure 14 shows the ACSS-D algorithm found a lower mean solution cost for nine out of the ten test files when compared with both the ACSS-Baseline (ACSS-B) and ACSS-Restart (ACSS-R) algorithms.

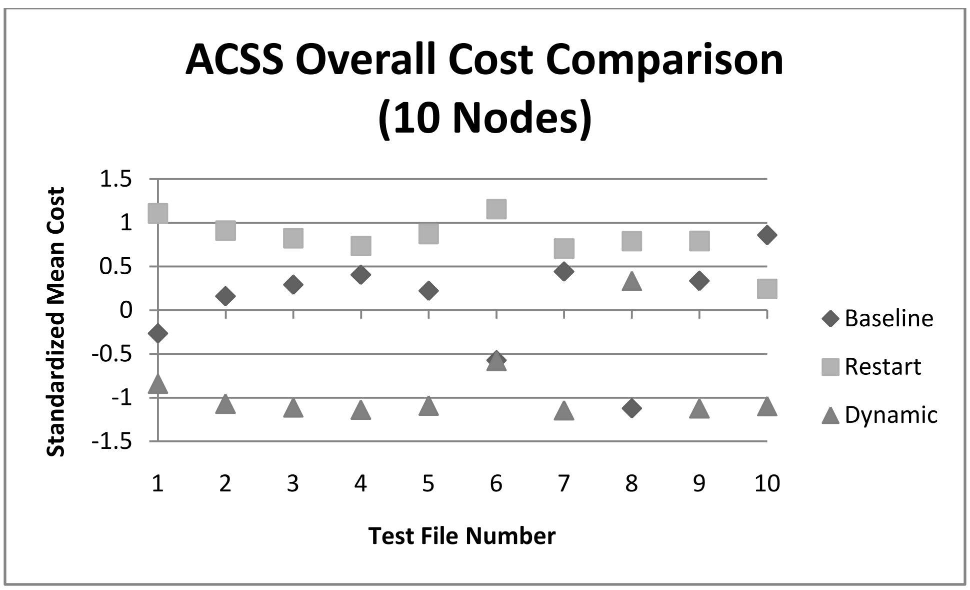


**Figure 14. ACSS Overall Cost Comparison by File (10 Nodes)**

Figure 15 shows the same data, but is averaged by the percentage change across each of the ten test files. ACSS-D had the lowest cost for each of the percentage changes except for the 10 percent change.

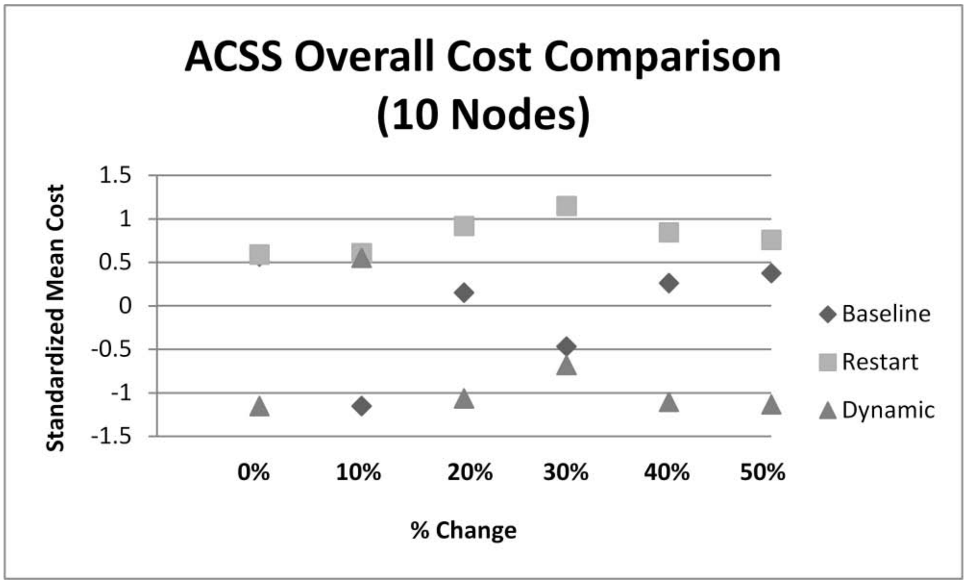


**Figure 15. ACSS Overall Cost Comparison by Percentage Change (10 Nodes)**

For the 15-node tests, the ACSS-D had the lowest mean cost per test file for all ten 15-node tests. Figure 16 shows the dynamic algorithm producing lower cost solutions than the ACSS-B and ACSS-R algorithms. Figure 17 also shows the same data for the 15-node problem, but is averaged by the percentage change across each of the ten test files. ACSS-D had the lowest cost for all six change percentages.

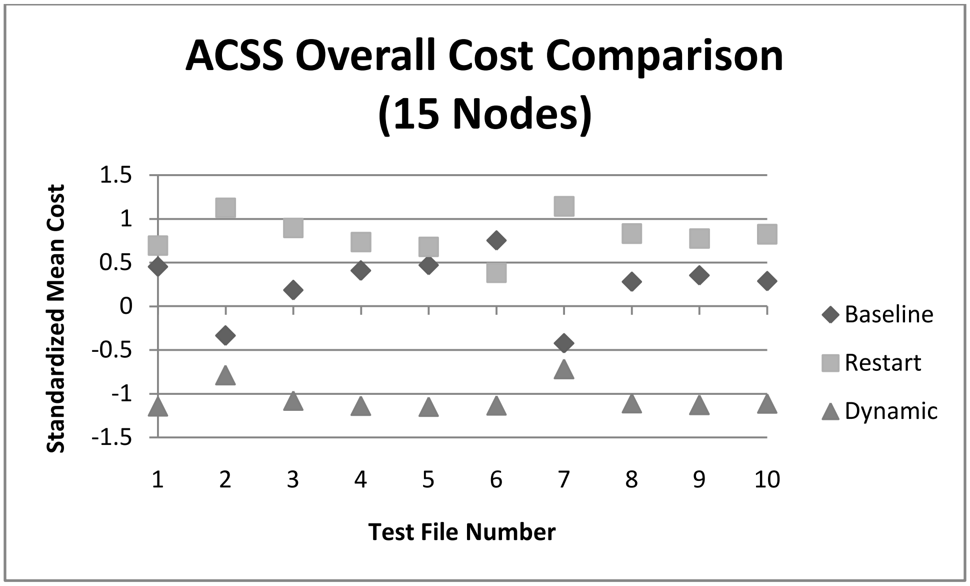


**Figure 16. ACSS Overall Cost Comparison by File (15 Nodes)**

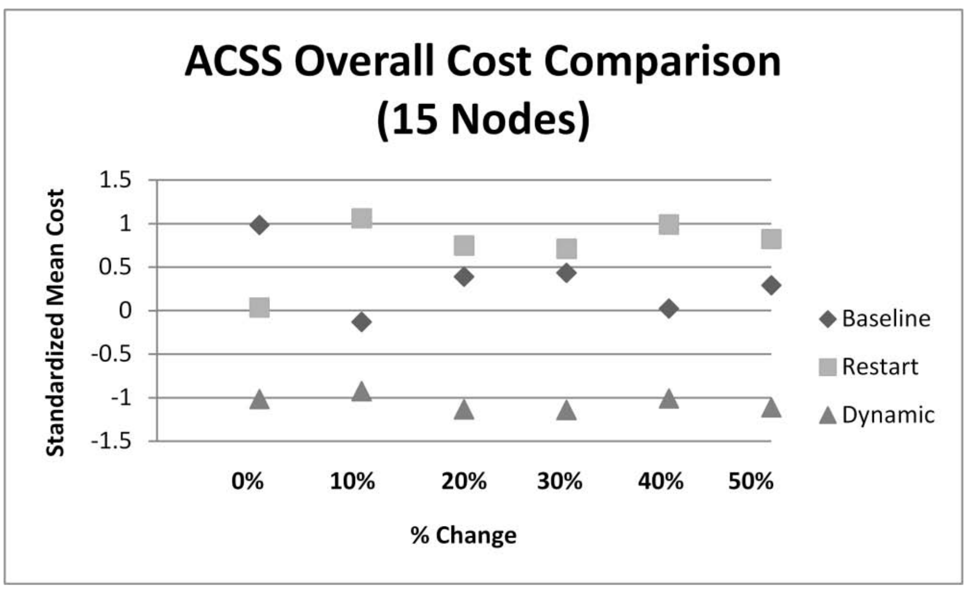


**Figure 17. ACSS Overall Cost Comparison by Percentage Change (15 Nodes)**

### 4.3.2 Dynamic Results - Ant Colony System Estimated (ACSE)

The ACSE-D had the lowest mean cost for six of the ten test files compared with the ACSE-B and ACSE-R approaches. Figure 18 illustrates the comparison of the three categories. Figure 19 displays the same data, but is averaged by the percentage change across each of the ten test files. ACSS-D had the lowest cost for all six change percentages.

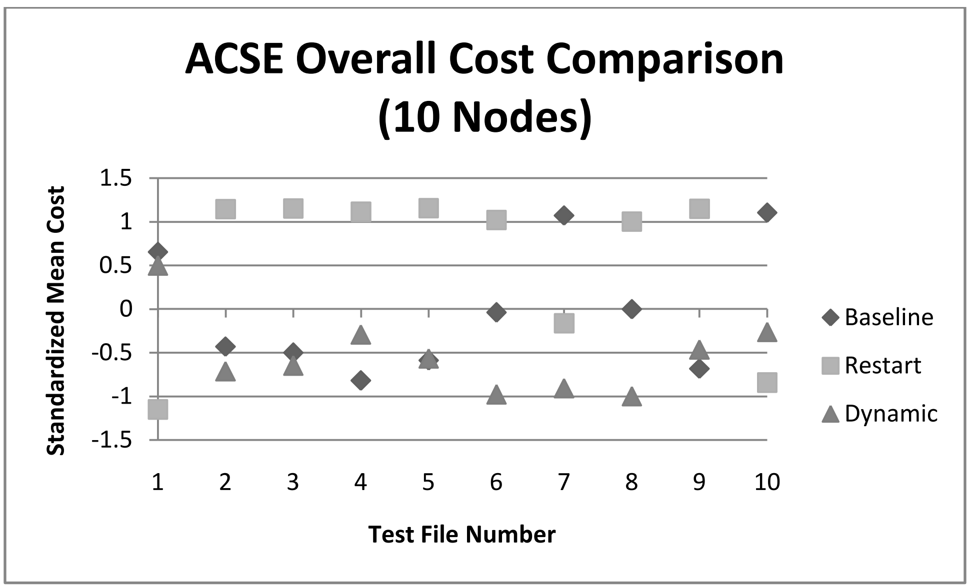


**Figure 18. ACSE Overall Cost Comparison by File (10 Nodes)**

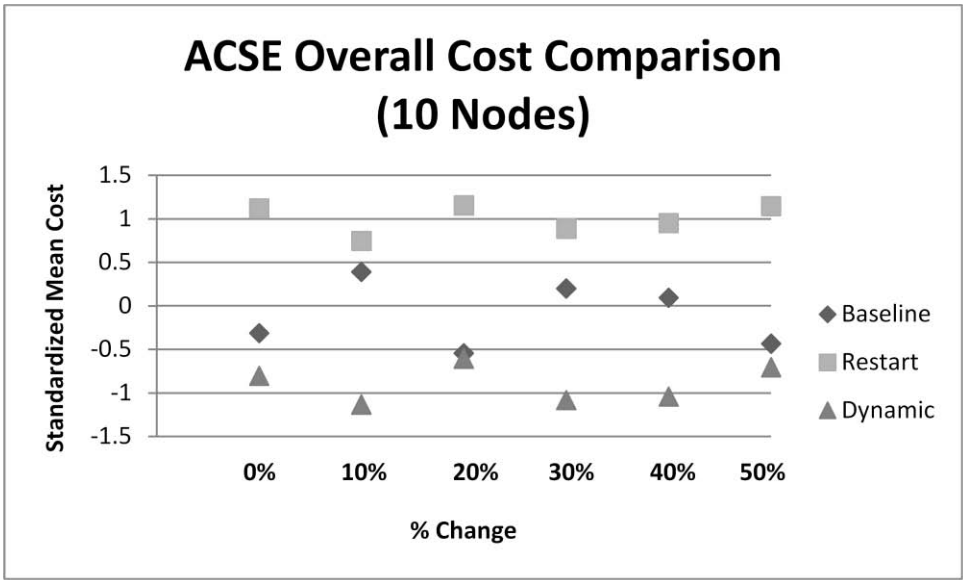


**Figure 19. ACSE Overall Cost Comparison by Percentage Change (10 Nodes)**

The ACSE-D had the lowest overall mean cost for seven of the ten 15-node test files. Figure 20 charts the overall cost comparison of the three dynamic algorithms. Figure 21 shows the same data, but is averaged by the percentage change across each of the ten test files. ACSS-D only had the lowest cost for two of the six change percentages.

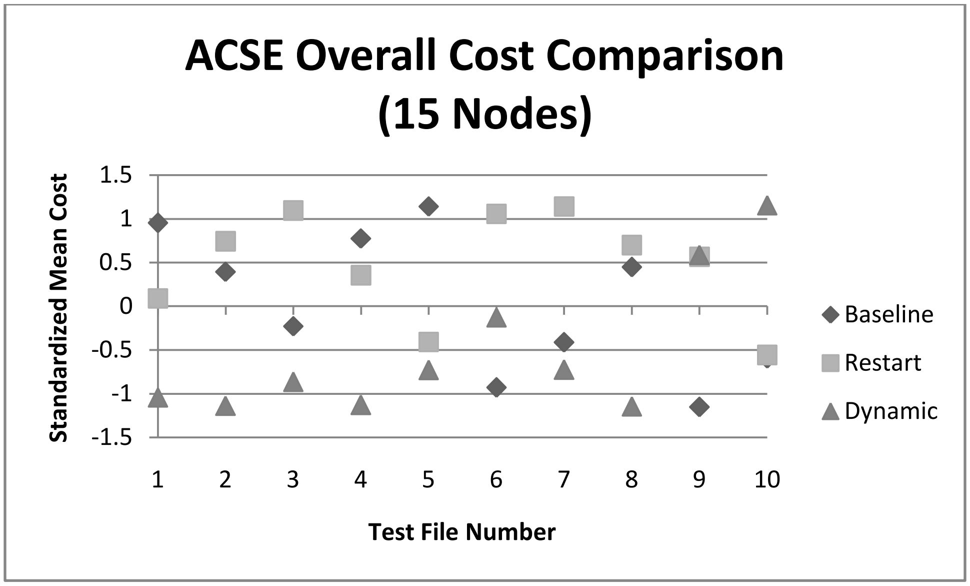


**Figure 20. ACSE Overall Cost Comparison by File (15Nodes)**

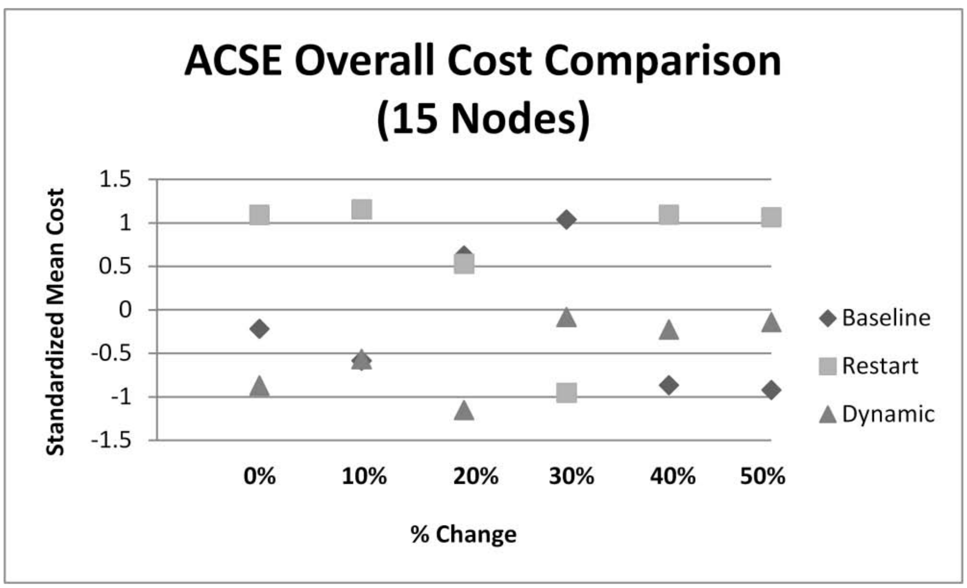


**Figure 21. ACSE Overall Cost Comparison by Percentage Change (15Nodes)**

### 4.3.3 ACSS Dynamic Convergence - 10 Nodes

The ACSS-D responds to strong dynamics within the 10-node network and is capable of adapting quickly to network change and converge on an acceptable solution.

The following graphs show each of the three categories (baseline, restart, and dynamic) responding to network change ranging from 10 percent up to 50 percent for a 10-node network. All three algorithms converge rather quickly. However, the ACSS-R algorithm knows the change is occurring and serves as a comparison algorithm. The ACSS-B algorithm does not respond as quickly and suffers from higher cost solutions when a change is introduced. Although the ACSS-D algorithm has no direct knowledge of a change being introduced, it is constantly evaluating its solutions and adjusting its exploration and exploitation ability. Therefore, the ACSS-D algorithm is dramatically faster at converging on a better solution, especially when change is encountered. For instance, Figure 22 through Figure 26 shows how the ACSS-B has a greater tendency to surge upwards when a change is introduced. However, the ACSS-D algorithm is more aware of change occurring. The ACCS-D algorithm encounters the change and responds quickly and almost immediately converges on to a new best topology solution. The graphs illustrate how quickly the ACSS-D algorithm flattens out as change is introduced.

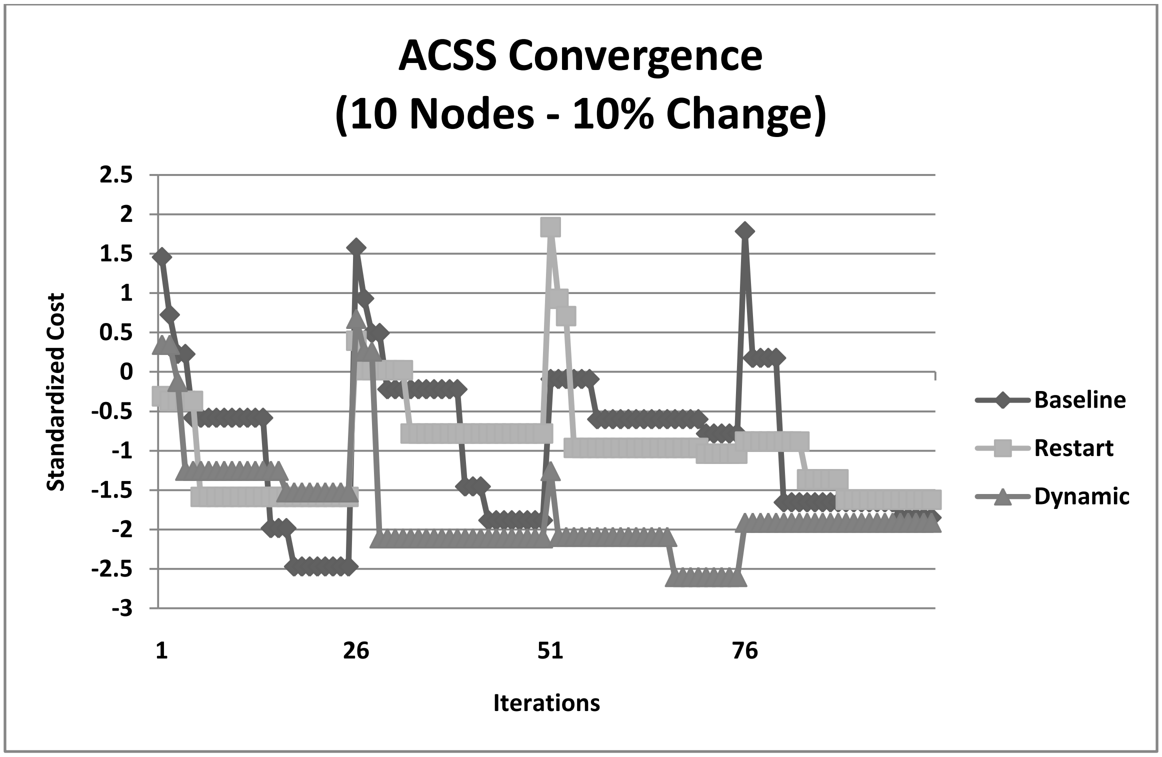


**Figure 22. ACSS Convergence - 10% Change (10 Nodes)**

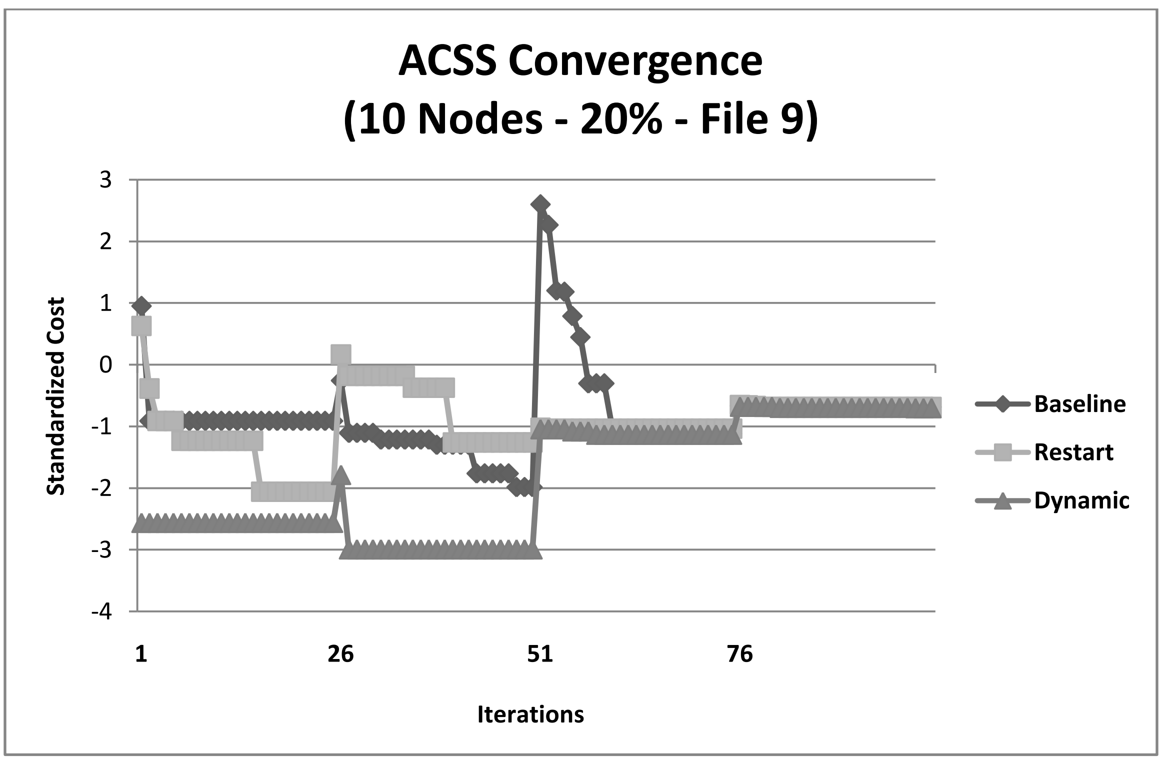


**Figure 23. ACSS Convergence - 20% Change (10 Nodes)**

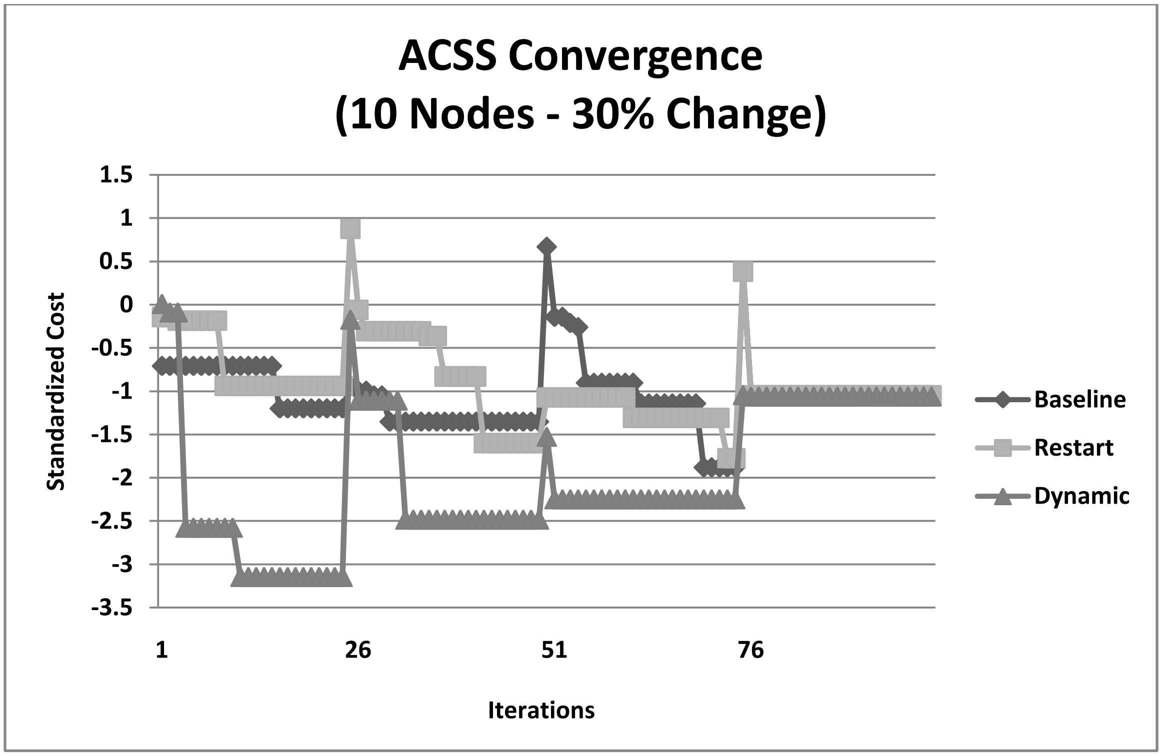


**Figure 24. ACSS Convergence - 30% Change (10 Nodes)**

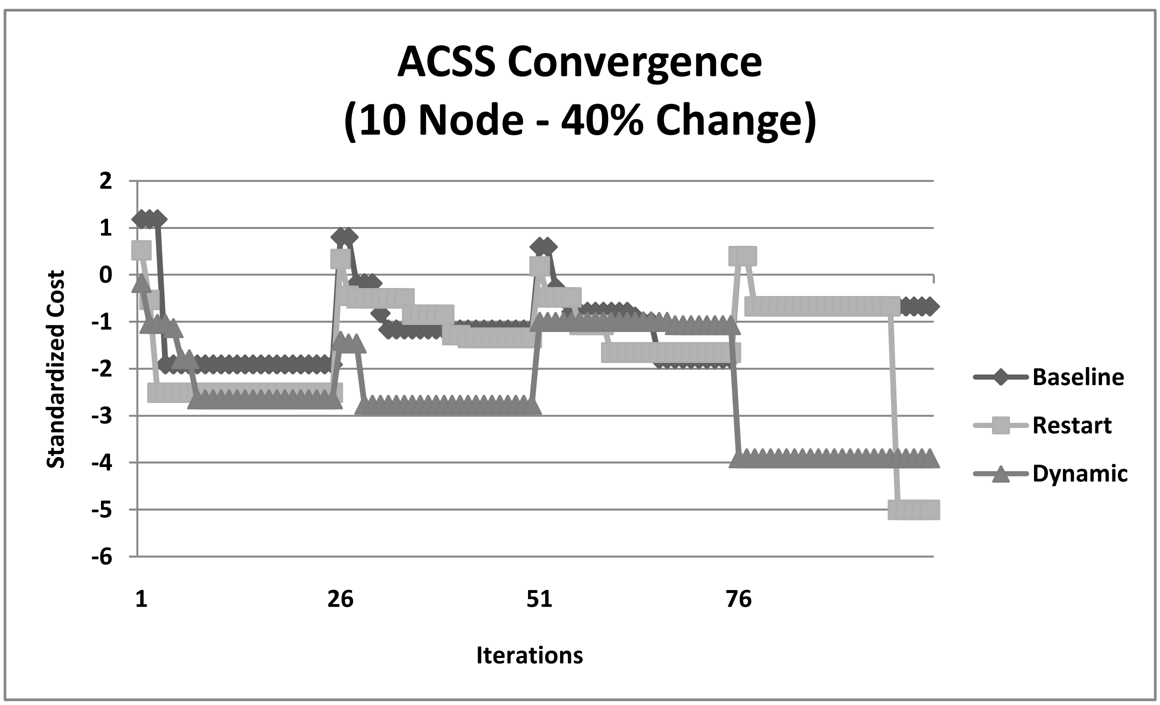


**Figure 25. ACSS Convergence - 40% Change (10 Nodes)**

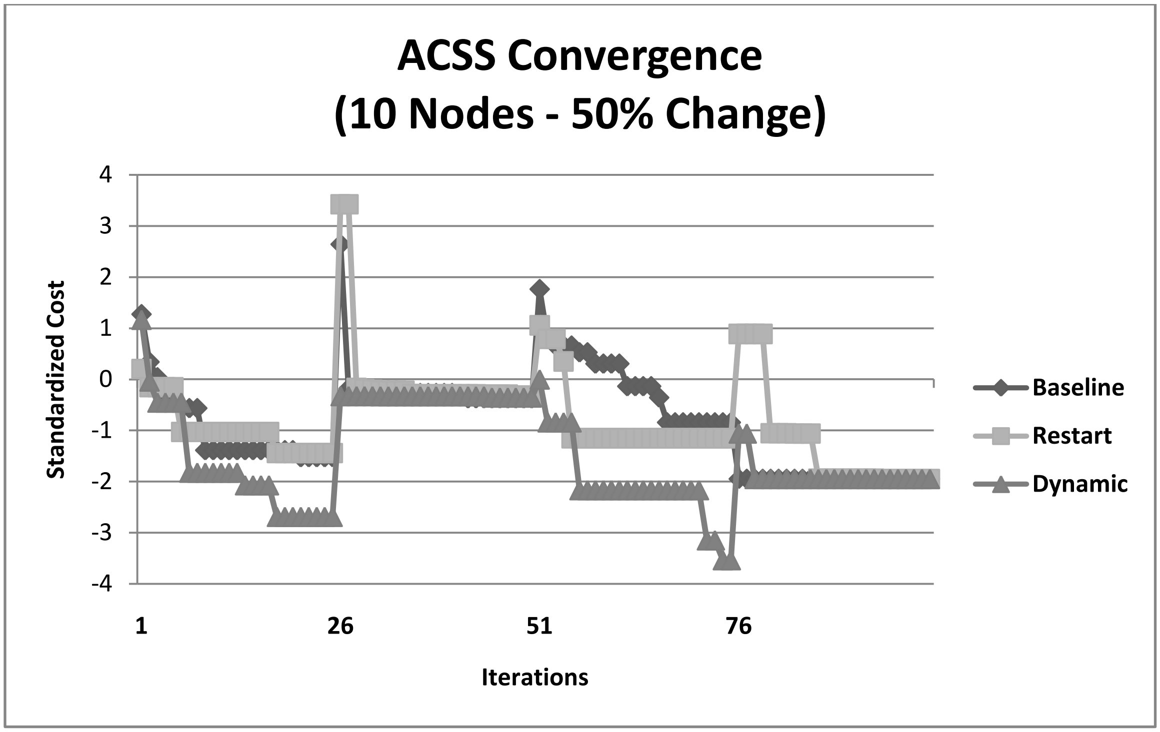


**Figure 26. ACSS Convergence - 50% Change (10 Nodes)**

### 4.3.4 ACSS Dynamic Convergence - 15 Nodes

The ACSS-D also responds to strong dynamics within the 15-node network and is capable of adapting rapidly to network change and converge on an acceptable solution. The following graphs illustrate how each of the three categories (ACSS-B, ACSS-R, and ACSS-D) respond to network change ranging from 10 percent up to 50 percent in a 15-node network. The 15-Node convergence graphs may not be as dramatic as the 10-node results. The primary reason is the 10-node ACSS tests had 30 iterations compared to only 10 iterations for the 15-node ACSS, generating three times the sample data. Similar to the 10-node graphs, with the 15-Node graphs (Figure 27 through Figure 31) DACSS-D continues to handle the change and quickly converges (flattens) towards a best solution.

The ACSS-B suffers from finding an initially high solution when change is encountered and is therefore slower to converge.

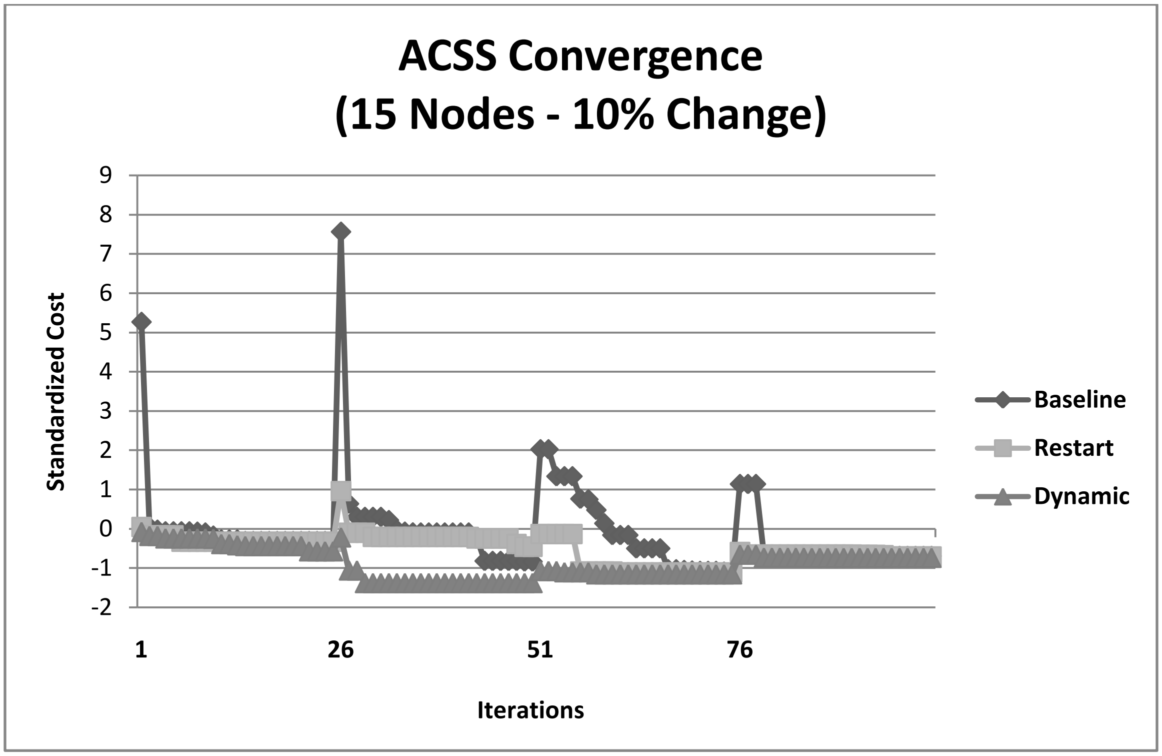


**Figure 27. ACSS Convergence - 10% Change (15 Nodes)**

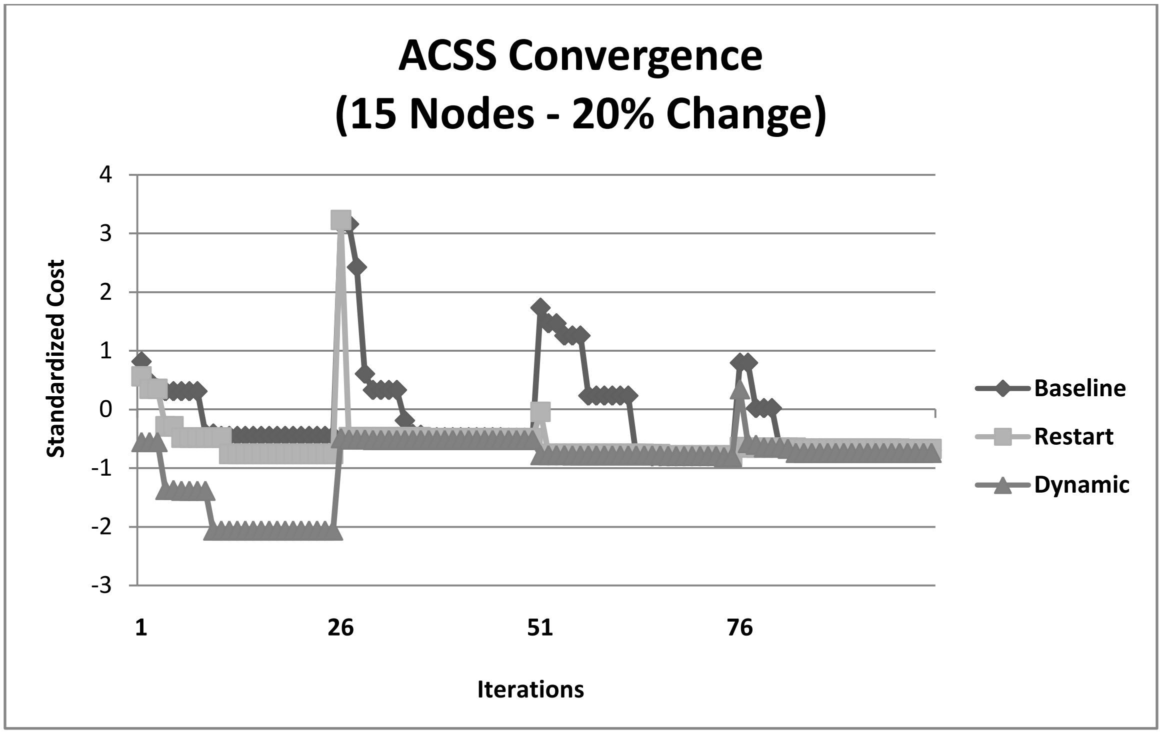


**Figure 28. ACSS Convergence - 20% Change (15 Nodes)**

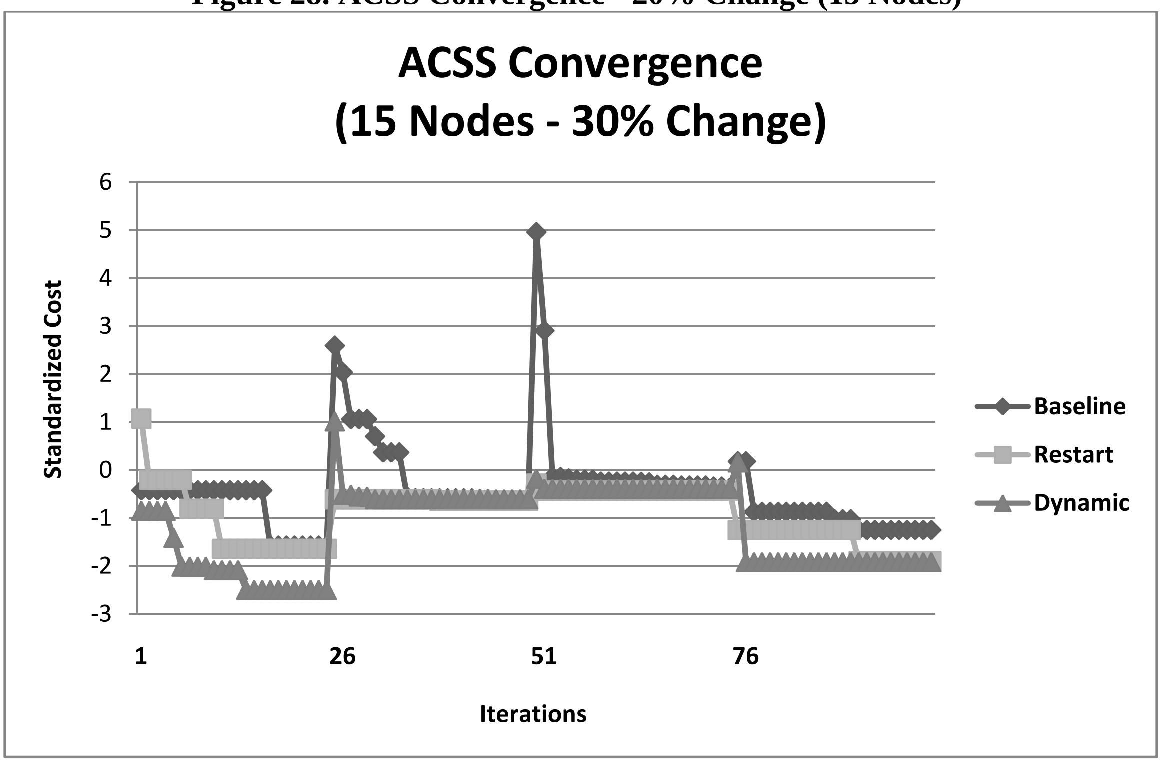


**Figure 29. ACSS Convergence - 30% Change (15 Nodes)**

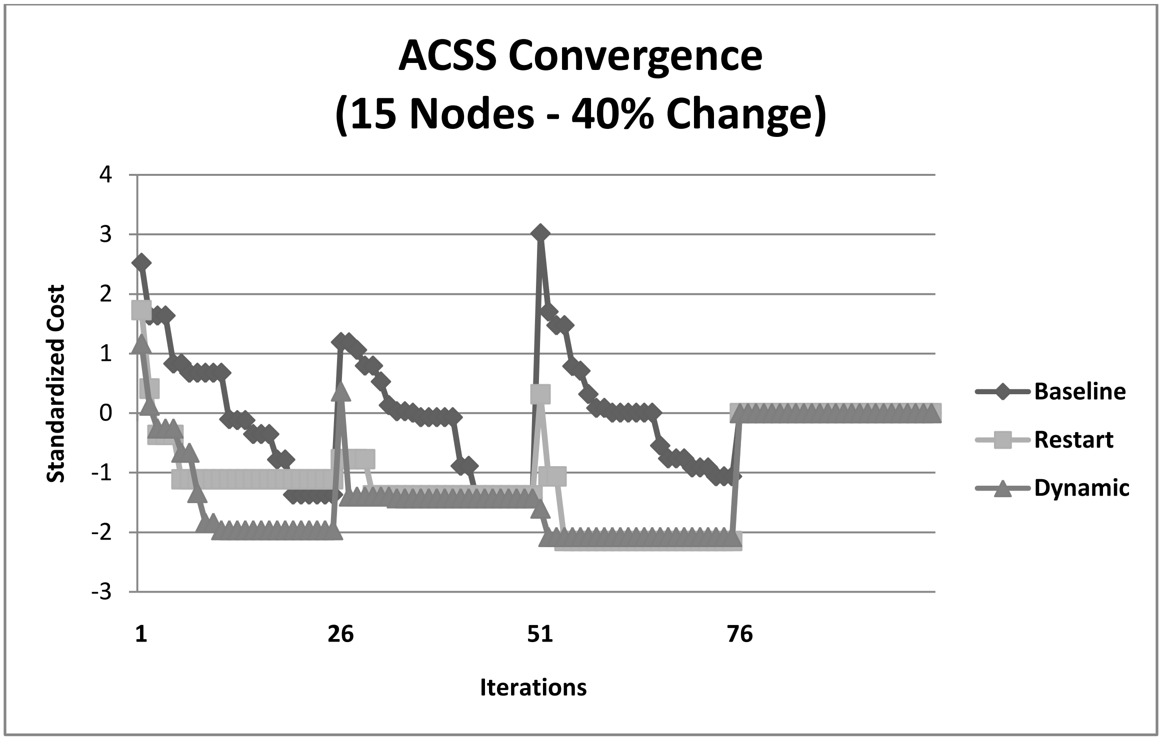


Figure 30. ACSS Convergence - 40% Change (15 Nodes)

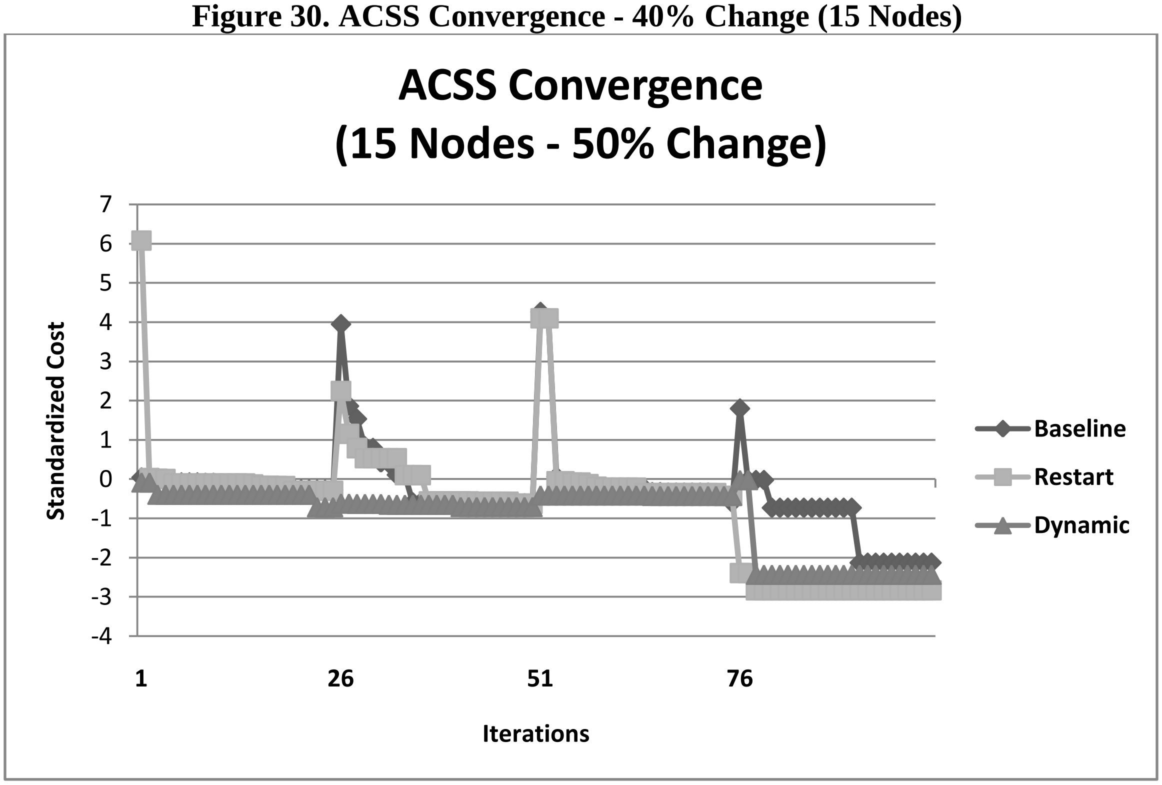


Figure 31. ACSS Convergence - 50% Change (15 Nodes)

### 4.4 Phase 3 Testing Results

The proposed algorithms not only need to be able to perform appropriately in a dynamic environment, but must also provide solutions in a timely manner. Phase 3 compares the algorithms tested in Phase 2 with their corresponding distributed approaches. The two features to be evaluated are solution costs and algorithm run time. The distributed approach must be able to construct similar topologies, at the same or reduced cost, and significantly reduce the amount of time required to construct the network topology. In addition, convergence graphs for the DACSS algorithms illustrate the convergence capability within a dynamic environment.

The Phase 3 testing compares the ACSS and ACSE centralized solvers with the DACSS and DACSE distributed solvers. The testing approach is similar to the dynamic, Phase 2. T-Tests are calculated to compare the solution results and establish if the means of both samples did in fact come from the same population. In addition, the computation time required by both the centralized and distributed solvers are compared and analyzed.

The solution costs generated in Phase 2 by the ACSS and the corresponding solution costs from DACSS are compared as are the Phase 2 ACSE and the Phase 3 DACSE solution costs. For these test comparison results to be considered statistically significant, it is unlikely it occurs by chance. A statistically significant difference between two sets of data results implies there is statistical evidence that a difference does exist. However, the actual difference is not known. In this research, two-tailed, two samples, assuming unequal variances *t*-tests about the means with a confidence level of 95% (alpha = 0.05) are used to determine the statistical significance of the results. The null hypothesis for all test is that the means come for the same population (i.e. the means

are equal). Also, the run times of the ACSS and DACSS algorithms and the run times of the ACSE and DACSE algorithms are compared. The results are displayed as a ratio of the distributed solver to the centralized solver. Full timing results are found in Appendix K.

### 4.4.1 Distributed ACSS (DACSS) - Cost Results (10 Nodes)

The DACSS generated cost solutions for a 10-node network are comparable to the ACSS approach for a 10-Node network. Two-tailed, *t*-tests were conducted for the solutions generated for each interval of each algorithm for all ten 10 files. For each test file, there are a total of 72 *t*-tests. Given that there are 10 test files, there are a total of 720 t-tests run. Overall, less than 19 percent of the tests indicated results where the null hypothesis that the means are equal would be rejected. Appendix H includes all the 10-node ACSS/DACSS *t*-test results. Table 9 provides a breakout by algorithm category of the number of *t*-tests that failed (had a p-value less than the alpha value of 0.05) for each file. For each algorithm, there were 24 *t*-tests conducted per file.

**Table 9. DACSS/ACSS *t*-Tests 95% Confidence Level (10 Nodes)**

| DACSS/ACSS *t*-Tests 95% Confidence Level (10 Nodes) | | | | | |
|---|---|---|---|---|---|
| **File** | **1** | **2** | **3** | **4** | **5** |
| **METHOD** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** |
| Baseline | 2/24 | 3/24 | 2/24 | 6/24 | 9/24 |
| Restart | 6/24 | 3/24 | 3/24 | 5/24 | 10/24 |
| Dynamic | 2/24 | 3/24 | 1/24 | 3/24 | 8/24 |
| **File** | **6** | **7** | **8** | **9** | **10** |
| **METHOD** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** |
| Baseline | 2/24 | 5/24 | 10/24 | 3/24 | 7/24 |
| Restart | 1/24 | 2/24 | 6/24 | 2/24 | 6/24 |
| Dynamic | 4/24 | 3/24 | 10/24 | 4/24 | 5/24 |

Complete ACSS and DACSS test results are included in Appendix C and Appendix E. Table 10 shows the mean cost ratio of DACSS and ACSS for each of the dynamic categories for all ten test files. Figure 32 illustrates this information graphically. A data point of 1 indicates the means for the distributed and centralized solvers are the same. In addition, a data point greater than 1 identifies the distributed solution is higher and a data point below 1 shows the distributed solution lower than the centralized approach. Clearly, the mean ratio is near 1.00 with an extremely low standard deviation for all three algorithms across all ten files. This is further support that the DACSS and ACSS produce very comparable results and over 80 percent of the time statistically equal means.

Table 10. DACSS/ACSS File Mean Cost Ratio Results (10 Nodes)

| DACSS/ACSS File Mean Cost Ratio Results - 10 Nodes | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| File | 1 | | 2 | | 3 | | 4 | | 5 | |
| METHOD | Mean Ratio | Std Dev | Mean Ratio | Std Dev | Mean Ratio | Std Dev | Mean Ratio | Std Dev | Mean Ratio | Std Dev |
| Baseline | 1.00 | 0.02 | 1.00 | 0.01 | 1.00 | 0.00 | 1.01 | 0.02 | 1.01 | 0.01 |
| Restart | 1.01 | 0.02 | 1.00 | 0.02 | 1.00 | 0.00 | 1.00 | 0.01 | 1.01 | 0.01 |
| Dynamic | 1.00 | 0.02 | 1.01 | 0.02 | 1.00 | 0.00 | 1.00 | 0.02 | 1.00 | 0.01 |
| File | 6 | | 7 | | 8 | | 9 | | 10 | |
| METHOD | Mean Ratio | Std Dev | Mean Ratio | Std Dev | Mean Ratio | Std Dev | Mean Ratio | Std Dev | Mean Ratio | Std Dev |
| Baseline | 1.00 | 0.00 | 1.00 | 0.02 | 1.00 | 0.01 | 1.00 | 0.00 | 1.00 | 0.00 |
| Restart | 1.00 | 0.00 | 1.00 | 0.01 | 1.00 | 0.01 | 1.00 | 0.00 | 1.00 | 0.00 |
| Dynamic | 1.00 | 0.01 | 1.00 | 0.01 | 1.01 | 0.01 | 1.00 | 0.01 | 1.00 | 0.01 |

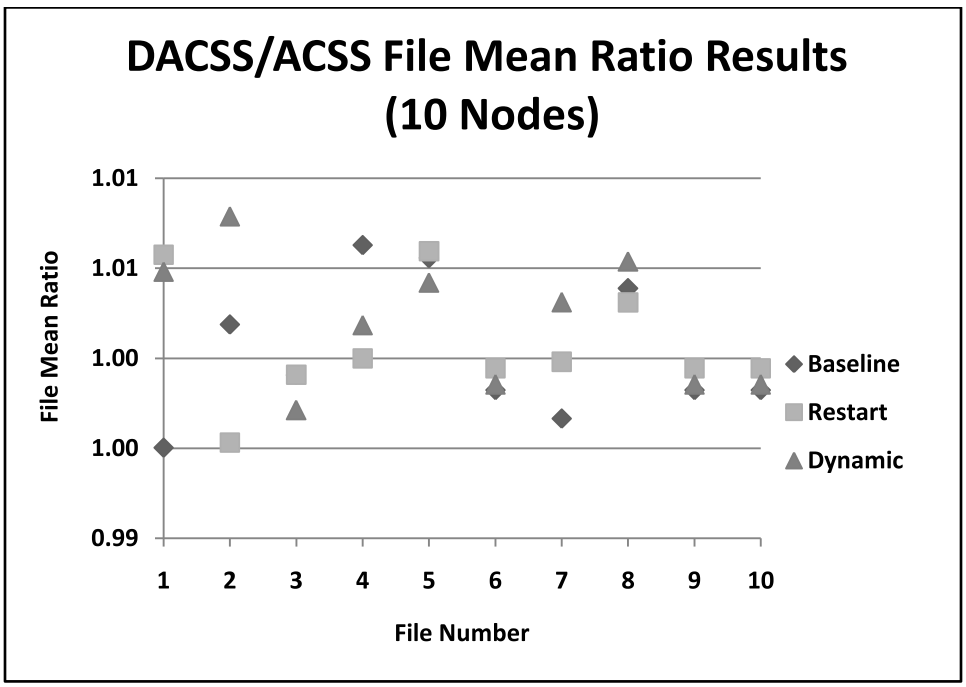


Figure 32. DACSS/ACSS File Mean Ratio Results (10 Nodes)

### 4.4.2 Distributed ACSE (DACSE) - Cost Results (10 Nodes)

The DACSE also generated cost solutions for a 10-node network comparable to the ACSE approach for a 10-Node network. Two-tailed, *t*-tests were conducted for the solutions generated (total of 720 *t*-tests). Overall, less than eight percent of the tests had a conclusion where the null hypothesis that the means are equal was rejected. Appendix G has all the 10-node ACSE/DACSE *t*-test results. Table 11 provides a breakout by algorithm category of the number of *t*-tests that failed (had a p-value less than the alpha value of 0.05) for each file. Again, for each algorithm, there were 24 *t*-tests conducted per file.

**Table 11. DACSE/ACSE *t*-Tests 95% Confidence Level (10 Nodes)**

| DACSE/ACSE *t*-Tests 95% Confidence Level (10 Nodes) | | | | | |
|---|---|---|---|---|---|
| **File** | **1** | **2** | **3** | **4** | **5** |
| **METHOD** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** |
| Baseline | 1/24 | 2/24 | 2/24 | 2/24 | 2/24 |
| Restart | 2/24 | 3/24 | 2/24 | 0/24 | 3/24 |
| Dynamic | 0/24 | 0/24 | 0/24 | 0/24 | 5/24 |
| **File** | **6** | **7** | **8** | **9** | **10** |
| **METHOD** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** |
| Baseline | 3/24 | 2/24 | 0/24 | 2/24 | 4/24 |
| Restart | 1/24 | 3/24 | 1/24 | 2/24 | 3/24 |
| Dynamic | 2/24 | 3/24 | 4/24 | 0/24 | 3/24 |

Complete ACSE and DACSE 10-node test results are also included in Appendix C and Appendix E. Table 12 shows the mean cost ratio of DACSE and ACSE for each of the dynamic categories for all ten test files. Figure 33 graphically displays this information. This information shows how the means of the ACSE and DACSE solutions are very comparable. Although the mean ratio is focused near 1, there are a few examples of a

higher standard deviation. The heuristics incorporated into the DACSE and ACSE solutions contributed to this fact. A more accurate heuristic would provide more consistent results and hence a lower standard deviation. Regardless though, the *t*-tests conclude that the two approaches have equivalent means over 90 percent of the time.

**Table 12. DACSE/ACSE File Mean Cost Ratio Results (10 Nodes)**

| DACSE/ACSE File Mean Cost Ratio Results - 10 Nodes | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **File** | **1** | | **2** | | **3** | | **4** | | **5** | |
| **METHOD** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** |
| Baseline | 1.00 | 0.05 | 0.99 | 0.15 | 1.07 | 0.23 | 1.03 | 0.09 | 1.00 | 0.04 |
| Restart | 1.02 | 0.11 | 0.97 | 0.16 | 1.24 | 0.48 | 1.01 | 0.09 | 1.00 | 0.04 |
| Dynamic | 0.99 | 0.11 | 1.00 | 0.08 | 0.99 | 0.22 | 1.02 | 0.08 | 1.01 | 0.04 |
| **File** | **6** | | **7** | | **8** | | **9** | | **10** | |
| **METHOD** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** |
| Baseline | 1.06 | 0.25 | 1.02 | 0.12 | 1.07 | 0.42 | 1.01 | 0.05 | 1.03 | 0.11 |
| Restart | 1.13 | 0.27 | 0.97 | 0.16 | 1.02 | 0.27 | 0.94 | 0.16 | 1.07 | 0.23 |
| Dynamic | 1.06 | 0.21 | 0.96 | 0.17 | 1.03 | 0.18 | 0.96 | 0.11 | 1.03 | 0.09 |

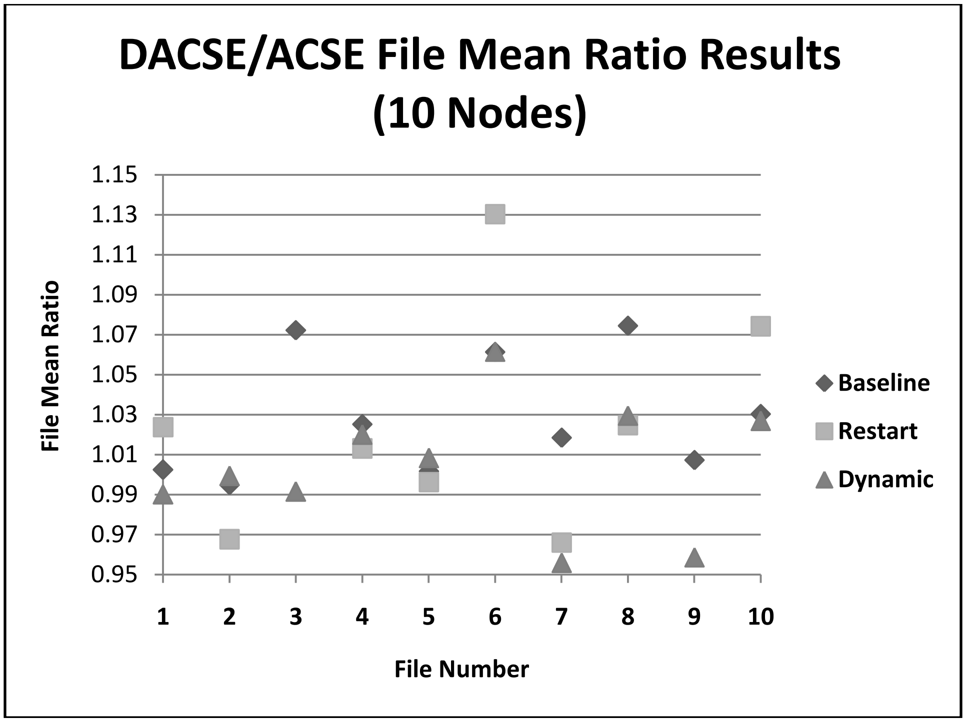


**Figure 33. DACSE/ACSE File Mean Ratio Results (10 Nodes)**

### 4.4.3 Distributed ACSS (DACSS) - Cost Results (15 Nodes)

The DACSS generated cost solutions that are statistically equivalent to the ACSS approach for a 15-Node network 80 percent of the time. Two-tailed, *t*-tests were conducted for the solutions generated (total of 720 *t*-tests). Overall, approximately 20 percent of the tests had a conclusion where the null hypothesis that the means are equal was rejected. Appendix J has all the 15-node ACSS/DACSS *t*-tests results. Table 13 provides a breakout by algorithm category of the number of *t*-tests that failed (had a p-value less than the alpha value of 0.05) for each file. For each algorithm, there were 24 *t*-tests conducted per file.

**Table 13. DACSS/ACSS *t*-Tests 95% Confidence Level (15 Nodes)**

| DACSS/ACSS *t*-Tests 95% Confidence Level (15 Nodes) | | | | | |
|---|---|---|---|---|---|
| **File** | **1** | **2** | **3** | **4** | **5** |
| **METHOD** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** |
| Baseline | 5/24 | 5/24 | 3/24 | 4/24 | 4/24 |
| Restart | 3/24 | 4/24 | 5/24 | 3/24 | 9/24 |
| Dynamic | 2/24 | 4/24 | 3/24 | 10/24 | 6/24 |
| **File** | **6** | **7** | **8** | **9** | **10** |
| **METHOD** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** |
| Baseline | 5/24 | 3/24 | 4/24 | 10/24 | 6/24 |
| Restart | 6/24 | 5/24 | 5/24 | 5/24 | 6/24 |
| Dynamic | 4/24 | 5/24 | 6/24 | 5/24 | 4/24 |

Complete ACSS and DACSS 15-node test results are included in Appendix D and Appendix F. Table 14 shows the mean cost ratio of DACSS and ACSS for each of the dynamic categories for all ten test files. The value of 1.0 serves as a reference point for the mean comparisons between the ACSS and DACSS algorithms. Figure 34 illustrates this information graphically. In addition to the *t*-test results, the mean ratio shows the values converge near 1. Most noteworthy, the DACSS-D algorithm is even below 1 for several of the files, providing a lower overall mean cost solution.

**Table 14. DACSS/ACSS File Mean Cost Ratio Results (15 Nodes)**

| DACSE/ACSE File Mean Cost Ratio Results - 15 Nodes | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **File** | **1** | | **2** | | **3** | | **4** | | **5** | |
| **METHOD** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** |
| Baseline | 1.00 | 0.04 | 1.01 | 0.03 | 1.00 | 0.06 | 1.01 | 0.04 | 1.01 | 0.05 |
| Restart | 0.99 | 0.03 | 1.00 | 0.04 | 0.99 | 0.06 | 1.01 | 0.07 | 0.99 | 0.11 |
| Dynamic | 1.00 | 0.02 | 1.00 | 0.03 | 1.01 | 0.07 | 1.01 | 0.04 | 0.99 | 0.08 |
| **File** | **6** | | **7** | | **8** | | **9** | | **10** | |
| **METHOD** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** |
| Baseline | 1.02 | 0.13 | 1.03 | 0.13 | 1.00 | 0.07 | 1.01 | 0.17 | 0.99 | 0.06 |
| Restart | 1.06 | 0.09 | 1.03 | 0.13 | 1.03 | 0.05 | 1.01 | 0.11 | 0.96 | 0.08 |
| Dynamic | 1.01 | 0.12 | 0.99 | 0.11 | 0.98 | 0.08 | 1.01 | 0.11 | 0.99 | 0.04 |

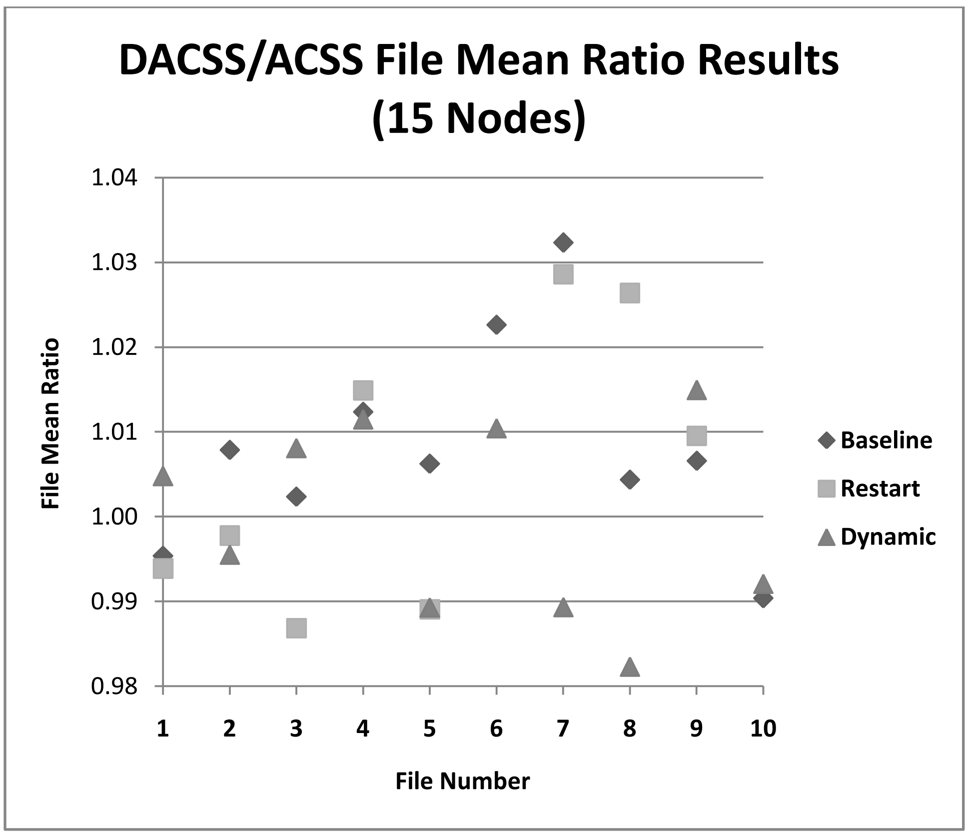


**Figure 34. DACSS/ACSS File Mean Ratio Results (15 Nodes)**

### 4.4.4 Distributed ACSE (DACSE) - Cost Results (15 Nodes)

The DACSE generated solutions that are statistically equivalent to the ACSE approach for a 15-Node network 95 percent of the time. Two-tailed, *t*-tests were conducted for the solutions generated (total of 720 *t*-tests). Overall, only 5 percent of the tests had a conclusion where the null hypothesis that the means are equal was rejected. Appendix I has all the 15-node ACSE/DACSE *t*-test results. Table 15 provides a breakout by algorithm category of the number of *t*-tests that failed (had a p-value less than the alpha value of 0.05) for each file. In addition, there were 24 *t*-tests conducted per algorithm per file.

**Table 15. DACSE/ACSE *t*-Tests 95% Confidence Level (15 Nodes)**

| DACSE/ACSE T-Tests 95% Confidence Level (15 Nodes) | | | | | |
|---|---|---|---|---|---|
| **File** | **1** | **2** | **3** | **4** | **5** |
| **METHOD** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** |
| Baseline | 0/24 | 2/24 | 1/24 | 1/24 | 1/24 |
| Restart | 0/24 | 1/24 | 1/24 | 4/24 | 2/24 |
| Dynamic | 1/24 | 2/24 | 0/24 | 3/24 | 2/24 |
| **File** | **6** | **7** | **8** | **9** | **10** |
| **METHOD** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** | **P-Value <Alpha** |
| Baseline | 0/24 | 2/24 | 2/24 | 2/24 | 2/24 |
| Restart | 0/24 | 2/24 | 1/24 | 0/24 | 0/24 |
| Dynamic | 0/24 | 2/24 | 2/24 | 2/24 | 1/24 |

Complete ACSE and DACSE 15-node test results are included in Appendix D and Appendix F. Table 16 shows the mean cost ratio of DACSS and ACSS for each of the dynamic categories for all ten test files. Figure 35 illustrates how each of the methods compares with their centralized counterparts. For the 15-node tests, the distributed solvers produce overall higher mean cost solutions, but are still extremely comparable

with the centralized approaches. Again, the distributed approach consistently produces similar and at times even better solutions than the centralized solvers. The distributed algorithms are generating the same quality of solution, but a fraction of the time.

**Table 16. DACSE/ACSE File Mean Cost Ratio Results (15 Nodes)**

| DACSE/ACSE File Mean Ratio Results - 15 Nodes | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **File** | **1** | | **2** | | **3** | | **4** | | **5** | |
| **METHOD** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** |
| Baseline | 1.02 | 0.15 | 0.99 | 0.15 | 1.07 | 0.11 | 1.01 | 0.12 | 1.00 | 0.17 |
| Restart | 0.99 | 0.16 | 0.95 | 0.11 | 1.02 | 0.15 | 1.05 | 0.06 | 1.01 | 0.19 |
| Dynamic | 1.02 | 0.17 | 1.03 | 0.17 | 1.00 | 0.13 | 1.01 | 0.12 | 1.06 | 0.23 |
| **File** | **6** | | **7** | | **8** | | **9** | | **10** | |
| **METHOD** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** | **Mean Ratio** | **Std Dev** |
| Baseline | 1.06 | 0.15 | 1.06 | 0.14 | 1.10 | 0.15 | 1.06 | 0.15 | 1.06 | 0.15 |
| Restart | 1.02 | 0.13 | 1.02 | 0.13 | 1.05 | 0.11 | 1.02 | 0.13 | 1.02 | 0.13 |
| Dynamic | 1.03 | 0.14 | 1.05 | 0.17 | 1.08 | 0.15 | 1.03 | 0.14 | 1.03 | 0.14 |

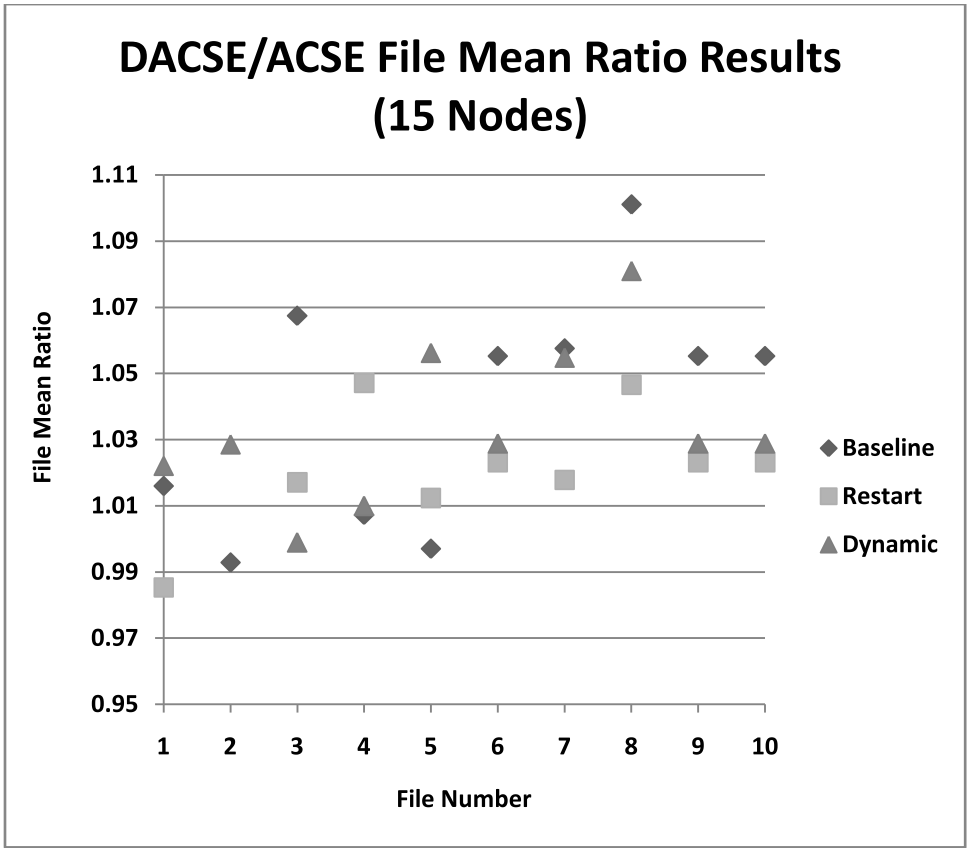


**Figure 35. DACSE/ACSE File Mean Ratio Results (15 Nodes)**

### 4.4.5 Distributed ACSS (DACSS) - Timing Results (10 Nodes)

The DACSS generated cost solutions for a 10-node network significantly faster than ACSS for a 10-Node network. Overall, the DACSS approach computes a solution on average in less than 17 percent of the time than ACSS. For each file/algorithm, the DACSS and ACSS mean was computed. Table 17 shows the ratio of the two means (DACSS/ACSS).

**Table 17. Ratio DACSS/ACSS (10 Nodes)**

| Timing Results - Ratio DACSS/ACSS (10 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.2378 | 0.0990 | 0.1084 | 0.1610 | 0.1866 | 0.1020 | 0.1081 | 0.2032 | 0.1928 | 0.1423 |
| **2** | 0.2446 | 0.1003 | 0.1053 | 0.1664 | 0.1909 | 0.1080 | 0.1166 | 0.2016 | 0.1986 | 0.1437 |
| **3** | 0.2394 | 0.0968 | 0.1086 | 0.1708 | 0.1926 | 0.1059 | 0.1128 | 0.2032 | 0.1892 | 0.1412 |
| **10%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.2346 | 0.1059 | 0.1182 | 0.1943 | 0.1865 | 0.1278 | 0.1157 | 0.1992 | 0.2561 | 0.1679 |
| **2** | 0.2334 | 0.1073 | 0.1142 | 0.1979 | 0.1924 | 0.1312 | 0.1138 | 0.2024 | 0.2527 | 0.1625 |
| **3** | 0.2428 | 0.1066 | 0.1184 | 0.1984 | 0.1852 | 0.1346 | 0.1182 | 0.2059 | 0.2475 | 0.1690 |
| **20%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.3130 | 0.1257 | 0.1604 | 0.2033 | 0.2218 | 0.1442 | 0.1340 | 0.2299 | 0.2159 | 0.1775 |
| **2** | 0.3147 | 0.1302 | 0.1664 | 0.1970 | 0.2197 | 0.1444 | 0.1347 | 0.2290 | 0.2135 | 0.1775 |
| **3** | 0.3117 | 0.1294 | 0.1664 | 0.1976 | 0.2222 | 0.1461 | 0.1322 | 0.2307 | 0.2112 | 0.1790 |
| **30%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.2548 | 0.1340 | 0.1350 | 0.1865 | 0.2065 | 0.1598 | 0.1558 | 0.2196 | 0.2403 | 0.1636 |
| **2** | 0.2551 | 0.1350 | 0.1425 | 0.1864 | 0.2057 | 0.1639 | 0.1568 | 0.2293 | 0.2457 | 0.1609 |
| **3** | 0.2588 | 0.1370 | 0.1381 | 0.1917 | 0.2075 | 0.1603 | 0.1584 | 0.2239 | 0.2439 | 0.1644 |
| **40%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.2093 | 0.1119 | 0.1397 | 0.1981 | 0.1969 | 0.1217 | 0.1300 | 0.2101 | 0.2371 | 0.1725 |
| **2** | 0.2049 | 0.1127 | 0.1400 | 0.2032 | 0.1989 | 0.1260 | 0.1300 | 0.2078 | 0.2352 | 0.1778 |
| **3** | 0.2091 | 0.1150 | 0.1373 | 0.2027 | 0.2005 | 0.1236 | 0.1296 | 0.2117 | 0.2366 | 0.1781 |
| **50%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.2248 | 0.1342 | 0.1478 | 0.1914 | 0.2102 | 0.2094 | 0.1677 | 0.2706 | 0.2278 | 0.1875 |
| **2** | 0.2263 | 0.1352 | 0.1478 | 0.1945 | 0.2085 | 0.2091 | 0.1673 | 0.2714 | 0.2263 | 0.1844 |
| **3** | 0.2027 | 0.1184 | 0.1309 | 0.1728 | 0.1824 | 0.1783 | 0.1454 | 0.2394 | 0.2026 | 0.1625 |

### 4.4.6 Distributed ACSE (DACSE) - Timing Results (10 Nodes)

The DACSE also generated cost solutions for a 10-node network considerably faster than ACSE for a 10-Node network. Overall, the DACSE approach constructed a solution in only 30 percent of the time that the ACSE approach needed. For each file/algorithm, the DACSE and ACSE mean was computed. Table 18 shows the ratio of the two means (DACSE/ACSE).

**Table 18. Ratio DACSE/ACSE (10 Nodes)**

| Timing Results - Ratio DACSE/ACSE (10 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0% | File 1 | File 2 | File 3 | File 4 | File 5 | File 6 | File 7 | File 8 | File 9 | File 10 |
| 1 | 0.3145 | 0.1554 | 0.1737 | 0.2398 | 0.2384 | 0.1871 | 0.2447 | 0.2714 | 0.2960 | 0.2170 |
| 2 | 0.3159 | 0.4510 | 0.1798 | 0.2435 | 0.2372 | 0.1920 | 0.2616 | 0.2817 | 0.2981 | 0.2232 |
| 3 | 0.3185 | 0.1623 | 0.1803 | 0.2428 | 0.2395 | 0.1925 | 0.2522 | 0.2798 | 0.3023 | 0.2260 |
| 10% | File 1 | File 2 | File 3 | File 4 | File 5 | File 6 | File 7 | File 8 | File 9 | File 10 |
| 1 | 0.3401 | 0.1848 | 0.2052 | 0.3166 | 0.2786 | 0.2209 | 0.3071 | 0.3069 | 0.3529 | 0.2752 |
| 2 | 0.3362 | 0.4204 | 0.1987 | 0.3086 | 0.2755 | 0.2135 | 0.3029 | 0.2980 | 0.3467 | 0.2699 |
| 3 | 0.3397 | 0.1860 | 0.2064 | 0.3162 | 0.2784 | 0.2198 | 0.3069 | 0.3068 | 0.3523 | 0.2767 |
| 20% | File 1 | File 2 | File 3 | File 4 | File 5 | File 6 | File 7 | File 8 | File 9 | File 10 |
| 1 | 0.4118 | 0.2098 | 0.2502 | 0.3059 | 0.3102 | 0.2431 | 0.2834 | 0.3408 | 0.3400 | 0.2775 |
| 2 | 0.4145 | 0.5739 | 0.2562 | 0.3108 | 0.3127 | 0.2487 | 0.3321 | 0.3481 | 0.3446 | 0.2831 |
| 3 | 0.4207 | 0.2187 | 0.2582 | 0.3195 | 0.3201 | 0.2535 | 0.2894 | 0.3490 | 0.3467 | 0.2841 |
| 30% | File 1 | File 2 | File 3 | File 4 | File 5 | File 6 | File 7 | File 8 | File 9 | File 10 |
| 1 | 0.4319 | 0.2681 | 0.2860 | 0.3576 | 0.3607 | 0.3350 | 0.3379 | 0.4051 | 0.4342 | 0.3222 |
| 2 | 0.4287 | 0.5222 | 0.2907 | 0.3547 | 0.3615 | 0.3296 | 0.3398 | 0.4051 | 0.4319 | 0.3225 |
| 3 | 0.4454 | 0.2794 | 0.3030 | 0.3677 | 0.3708 | 0.3397 | 0.3473 | 0.4193 | 0.4462 | 0.3313 |
| 40% | File 1 | File 2 | File 3 | File 4 | File 5 | File 6 | File 7 | File 8 | File 9 | File 10 |
| 1 | 0.3489 | 0.1899 | 0.2323 | 0.3842 | 0.2937 | 0.2564 | 0.3124 | 0.4207 | 0.3697 | 0.3019 |
| 2 | 0.3584 | 0.4795 | 0.2563 | 0.3455 | 0.3287 | 0.2589 | 0.3207 | 0.3735 | 0.3840 | 0.3076 |
| 3 | 0.3592 | 0.2142 | 0.2596 | 0.3488 | 0.3279 | 0.2604 | 0.3133 | 0.4071 | 0.3509 | 0.3072 |
| 50% | File 1 | File 2 | e 3 | File 4 | File 5 | File 6 | File 7 | File 8 | File 9 | File 10 |
| 1 | 0.3500 | 0.2300 | 0.2506 | 0.3022 | 0.3164 | 0.3304 | 0.3157 | 0.3953 | 0.3685 | 0.2884 |
| 2 | 0.3392 | 0.3982 | 0.2417 | 0.2946 | 0.3072 | 0.3189 | 0.3419 | 0.3876 | 0.3404 | 0.2804 |
| 3 | 0.3491 | 0.2293 | 0.2499 | 0.3023 | 0.3148 | 0.3306 | 0.3154 | 0.3961 | 0.3683 | 0.2874 |

### 4.4.7 Distributed ACSS (DACSS) - Timing Results (15 Nodes)

The DACSS generated cost solutions for a 15-node network drastically faster than ACSS for a 15-Node network. Overall, the DACSS approach computes a solution in less than 13 percent of the time compared with the ACSS approach. For each file/algorithm, the DACSS and ACSS mean was computed. Table 19 shows the ratio of the two means (DACSS/ACSS).

Table 19. Ratio DACSS/ACSS (15 Nodes)

| Timing Results - Ratio DACSE/ACSE (15 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.0680 | 0.0949 | 0.1004 | 0.1097 | 0.0896 | 0.1269 | 0.1505 | 0.0823 | 0.0864 | 0.0858 |
| **2** | 0.0702 | 0.0821 | 0.1000 | 0.1176 | 0.0799 | 0.1509 | 0.1633 | 0.0748 | 0.0752 | 0.0890 |
| **3** | 0.0752 | 0.0833 | 0.1055 | 0.1154 | 0.0840 | 0.1580 | 0.1695 | 0.0788 | 0.0778 | 0.1047 |
| **10%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.0790 | 0.0874 | 0.1073 | 0.1146 | 0.0922 | 0.1542 | 0.1296 | 0.0896 | 0.0878 | 0.0977 |
| **2** | 0.0669 | 0.0788 | 0.0968 | 0.1123 | 0.0818 | 0.1268 | 0.1295 | 0.0768 | 0.0778 | 0.0940 |
| **3** | 0.0834 | 0.0880 | 0.1084 | 0.1215 | 0.0944 | 0.1584 | 0.1514 | 0.0886 | 0.0880 | 0.1035 |
| **20%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.0809 | 0.1039 | 0.1025 | 0.1206 | 0.1088 | 0.1450 | 0.1606 | 0.0867 | 0.0882 | 1.0470 |
| **2** | 0.0798 | 0.1043 | 0.1052 | 0.1243 | 0.1134 | 0.1501 | 0.1657 | 0.0864 | 0.0848 | 1.0101 |
| **3** | 0.0821 | 0.0988 | 0.0952 | 0.1293 | 0.1112 | 0.1509 | 0.1647 | 0.0903 | 0.0892 | 1.0941 |
| **30%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.0832 | 0.1032 | 0.1247 | 0.1562 | 0.1022 | 0.1436 | 0.5203 | 0.0904 | 0.0847 | 0.0999 |
| **2** | 0.0852 | 0.1058 | 0.1210 | 0.1555 | 0.1027 | 0.1444 | 0.1983 | 0.0902 | 0.0850 | 0.1038 |
| **3** | 0.0862 | 0.1029 | 0.1225 | 0.1578 | 0.1003 | 0.1464 | 0.2106 | 0.0912 | 0.0857 | 0.1013 |
| **40%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.0945 | 0.0968 | 0.1266 | 0.1350 | 0.1080 | 0.1522 | 0.1507 | 0.0987 | 0.0936 | 0.1184 |
| **2** | 0.0931 | 0.0964 | 0.1241 | 0.1391 | 0.1032 | 0.1525 | 0.1460 | 0.0944 | 0.0974 | 0.1167 |
| **3** | 0.0934 | 0.0974 | 0.1236 | 0.1384 | 0.1047 | 0.1451 | 0.1501 | 0.0954 | 0.0945 | 0.1182 |
| **50%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.0974 | 0.1095 | 0.1258 | 0.1302 | 0.1024 | 0.1456 | 0.1774 | 0.0967 | 0.0884 | 0.1312 |
| **2** | 0.0949 | 0.1084 | 0.1278 | 0.1374 | 0.1041 | 0.1469 | 0.1818 | 0.0969 | 0.0875 | 0.1210 |
| **3** | 0.0972 | 0.1090 | 0.1261 | 0.1285 | 0.1037 | 0.1461 | 0.1805 | 0.1004 | 0.0915 | 0.1257 |

### 4.4.8 Distributed ACSE (DACSE) - Timing Results (15 Nodes)

The DACSE also generated cost solutions for a 10-node network considerably faster than ACSE for a 10-Node network. Overall, the DACSE approach constructed a solution in 18 percent of the time that the ACSE approach needed. For each file/algorithm, the DACSE and ACSE mean was computed. Table 20 shows the ratio of the two means (DACSE/ACSE).

**Table 20. Ratio DACSE/ACSE (15 Nodes)**

| Timing Results - Ratio DACSE/ACSE (15 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.1276 | 0.1432 | 0.1760 | 0.1974 | 0.1656 | 0.2058 | 0.1941 | 0.1336 | 0.1215 | 0.1610 |
| **2** | 0.1265 | 0.1425 | 0.1748 | 0.2007 | 0.1635 | 0.2023 | 0.1971 | 0.1339 | 0.1164 | 0.1613 |
| **3** | 0.1257 | 0.1431 | 0.1762 | 0.2010 | 0.1664 | 0.2040 | 0.1975 | 0.1319 | 0.1217 | 0.1596 |
| **10%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.1405 | 0.1543 | 0.1952 | 0.2096 | 0.1876 | 0.2312 | 0.2091 | 0.1643 | 0.1431 | 0.1773 |
| **2** | 0.1345 | 0.1522 | 0.1869 | 0.2006 | 0.1801 | 0.2210 | 0.1971 | 0.1597 | 0.1459 | 0.1772 |
| **3** | 0.1354 | 0.1533 | 0.1879 | 0.2011 | 0.1807 | 0.2215 | 0.2009 | 0.1586 | 0.1429 | 0.1702 |
| **20%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.1384 | 0.1699 | 0.1870 | 0.2093 | 0.1952 | 0.2243 | 0.2338 | 0.1559 | 0.1466 | 0.1625 |
| **2** | 0.1363 | 0.1597 | 0.1831 | 0.2041 | 0.1902 | 0.2199 | 0.2338 | 0.1539 | 0.1451 | 0.1531 |
| **3** | 0.1375 | 0.1572 | 0.1863 | 0.2088 | 0.1946 | 0.2245 | 0.2345 | 0.1554 | 0.1456 | 0.1588 |
| **30%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.1526 | 0.1815 | 0.2110 | 0.2540 | 0.1970 | 0.2439 | 0.2915 | 0.1738 | 0.1594 | 0.1920 |
| **2** | 0.1463 | 0.1745 | 0.2026 | 0.2438 | 0.1869 | 0.2326 | 0.2730 | 0.1662 | 0.1525 | 0.1829 |
| **3** | 0.1465 | 0.1752 | 0.2032 | 0.2447 | 0.1871 | 0.2335 | 0.2730 | 0.1665 | 0.1533 | 0.1836 |
| **40%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.1511 | 0.1608 | 0.2034 | 0.2324 | 0.1912 | 0.2386 | 0.2208 | 0.1649 | 0.1629 | 0.1917 |
| **2** | 0.1519 | 0.1616 | 0.2048 | 0.2337 | 0.1928 | 0.2388 | 0.2197 | 0.1663 | 0.1636 | 0.1931 |
| **3** | 0.1511 | 0.1570 | 0.2038 | 0.2291 | 0.1852 | 0.2323 | 0.2196 | 0.1652 | 0.1595 | 0.1918 |
| **50%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **1** | 0.1799 | 0.1823 | 0.2220 | 0.2325 | 0.1973 | 0.2362 | 0.2540 | 0.1791 | 0.1613 | 0.2107 |
| **2** | 0.1702 | 0.1733 | 0.2145 | 0.2239 | 0.1898 | 0.2276 | 0.2455 | 0.1772 | 0.1526 | 0.2004 |
| **3** | 0.1798 | 0.1843 | 0.2268 | 0.2361 | 0.1995 | 0.2392 | 0.2573 | 0.1822 | 0.1625 | 0.2109 |

### 4.4.9 Distributed ACSS (DACSS) - Convergence (10 Nodes)

The DACSS-D responds to strong dynamics within the 10-node network and is capable of adapting quickly to network change and converge on an acceptable solution similar to the dynamic ACSS algorithm. A series of graphs show each of the three dynamic distributed categories (baseline, restart, and dynamic) responding to network change ranging from 10 percent up to 50 percent. The DACSS-D algorithm again

converges quickly, adapting and responding to network change. The following graphs depict how the DACSS-D reacts at the beginning of each interval. The change in the network topology score is minimal when compared to the other approaches (DACSS-B and DACSS-R). In addition, DACSS-D still converges quickly on the “best” solution and is shown by the flat graph line. Figure 36 shows the convergence at 10 percent change up to Figure 40 for 50 percent change.

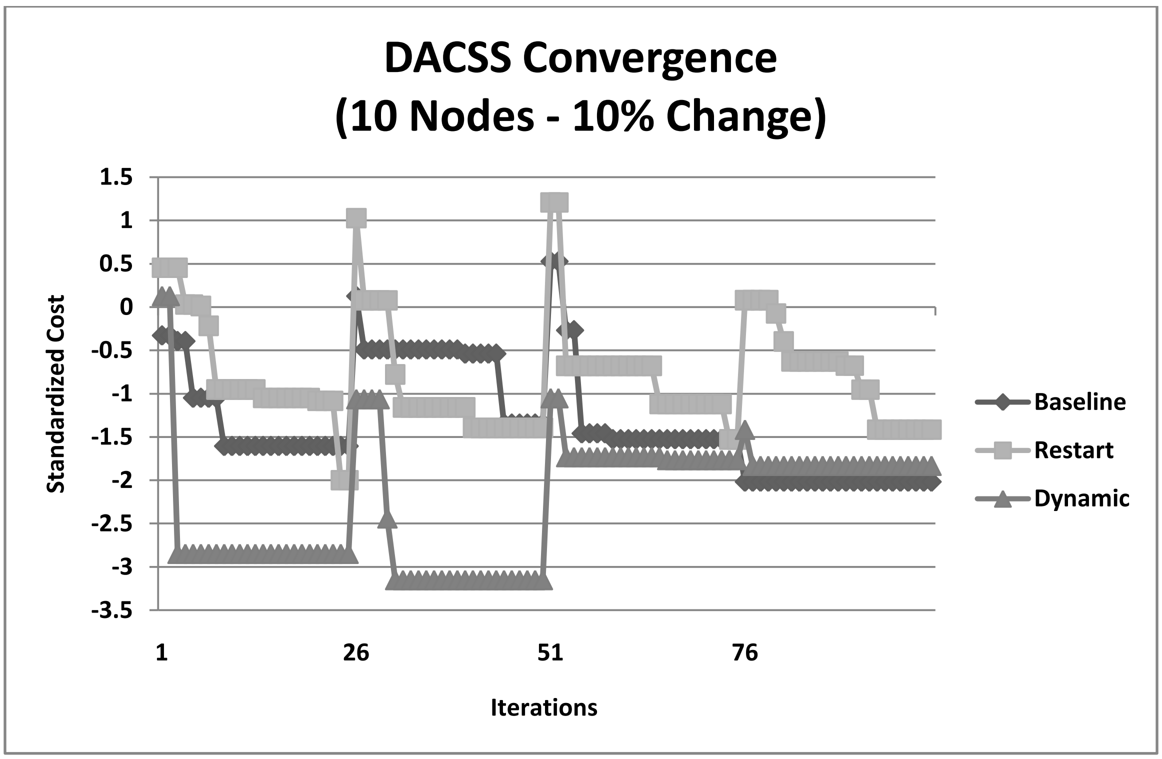


**Figure 36. DACSS Convergence - 10% Change (10 Nodes)**

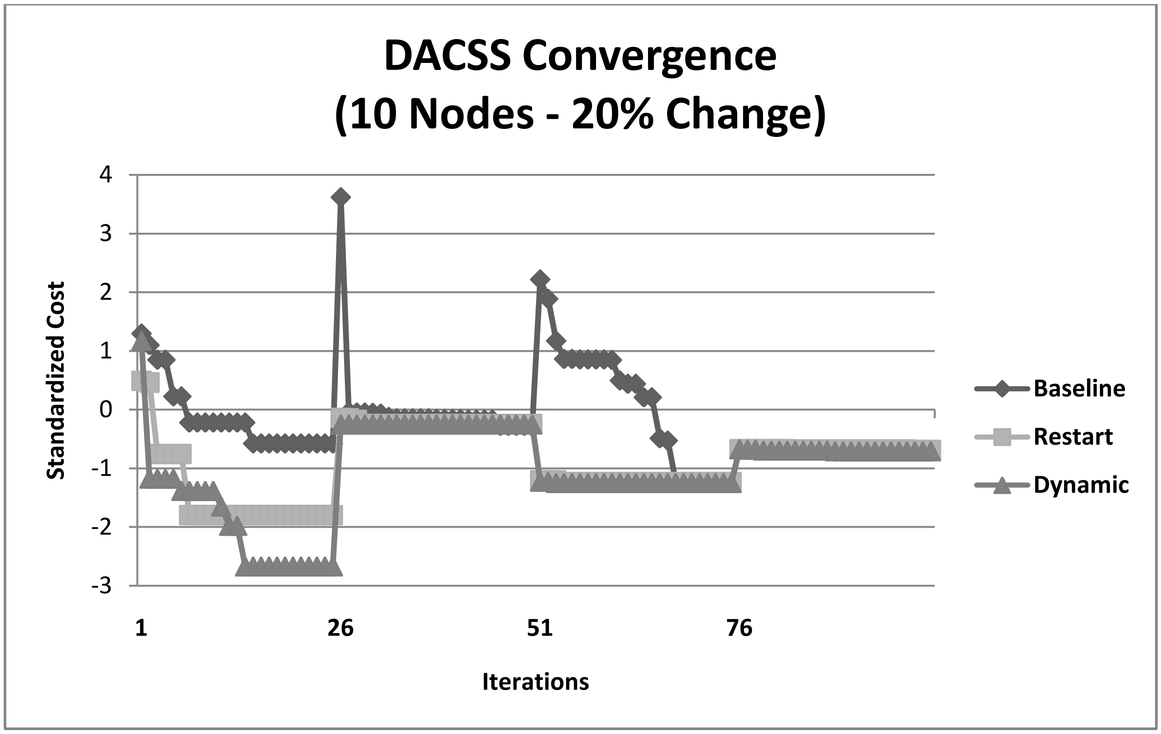


**Figure 37. DACSS Convergence - 20% Change (10 Nodes)**

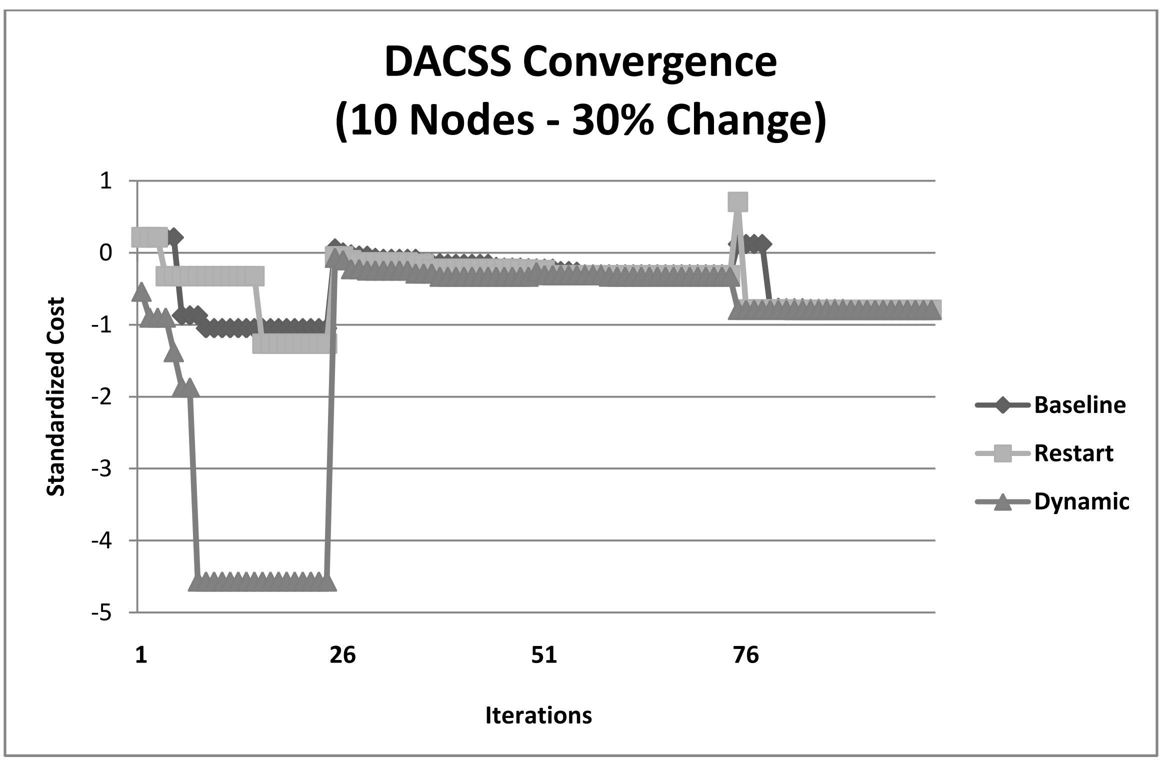


**Figure 38. DACSS Convergence - 30% Change (10 Nodes)**

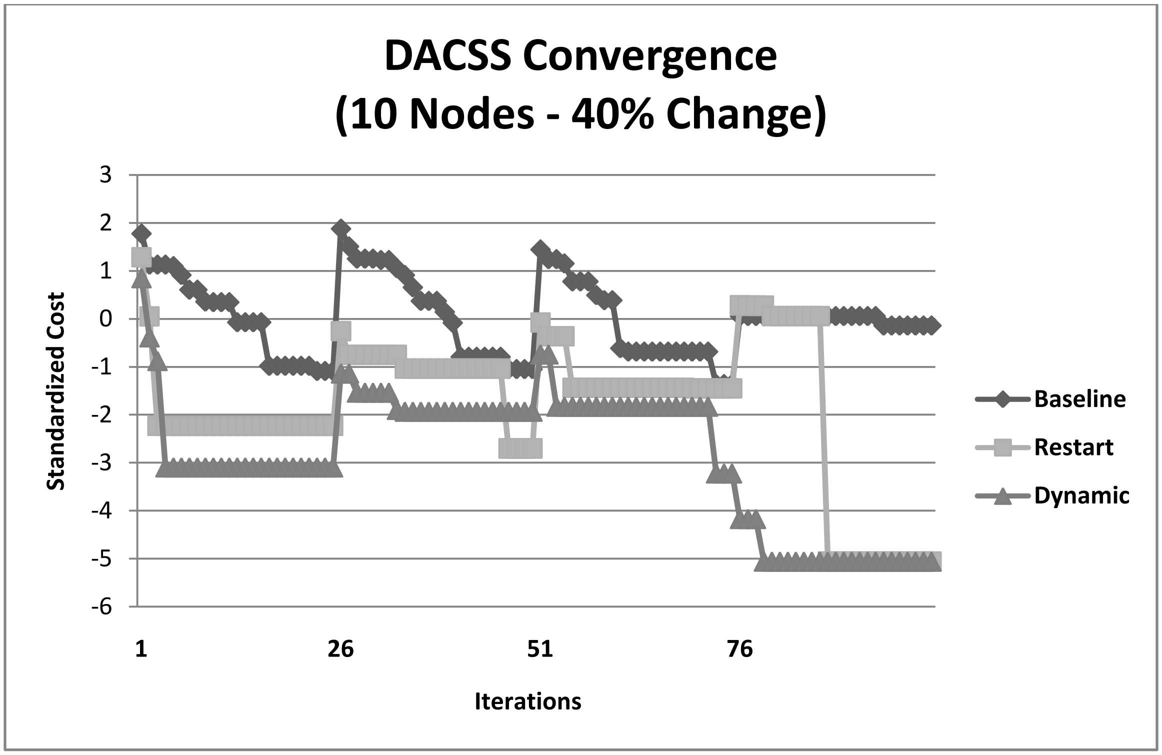


Figure 39. DACSS Convergence - 40% Change (10 Nodes)

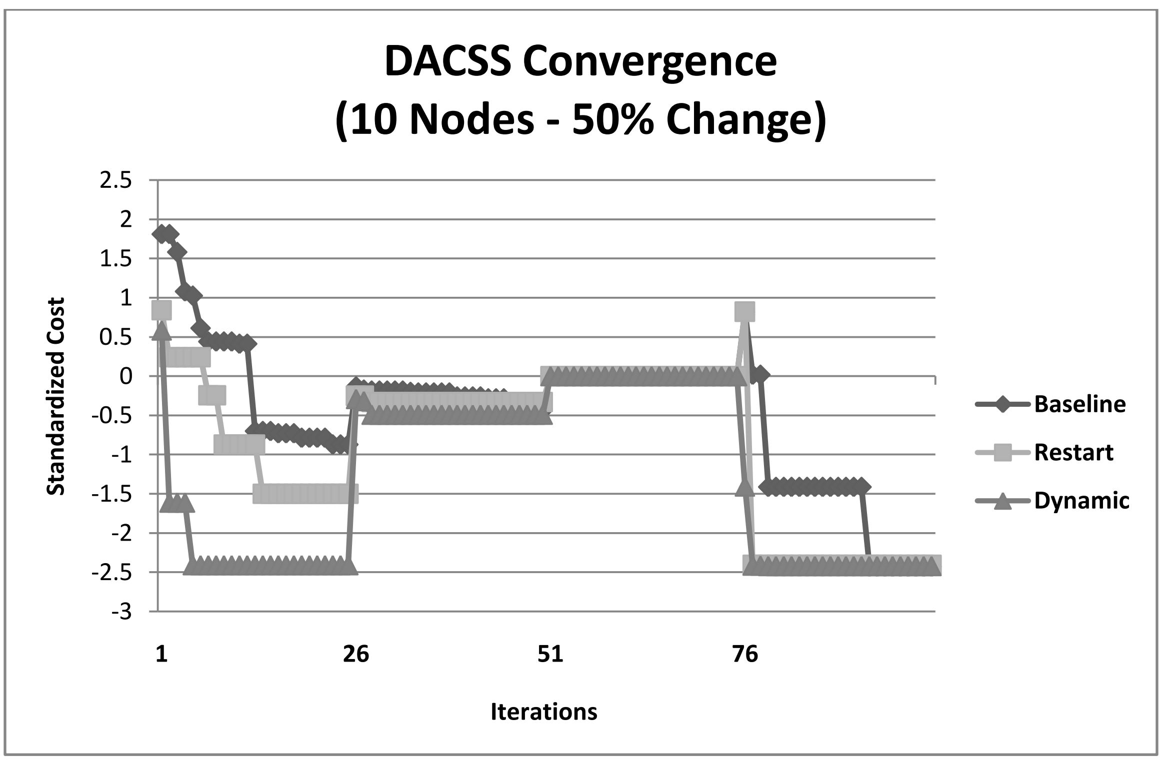


Figure 40. DACSS Convergence - 50% Change (10 Nodes)

### 4.4.10 Distributed ACSS (DACSS) - Convergence (15 Nodes)

The DACSS-D responds to strong dynamics within the 15-node network too. The algorithm converges quickly towards a solution and is also capable of adapting quickly to network change. The next set of graphs show each of the three dynamic distributed categories (baseline, restart, and dynamic) responding to network change ranging from 10 percent up to 50 percent in a 15-node network. With the 15-node network, testing was limited to only 10 trials per file. However, the DACSS-D still converges quickly. The DACSS-B also converges quickly, but tends to spike upwards when a change is introduced. DACSS-D is less affected by the change and shows a slight increase vice a large spike in the graph. The following charts (Figure 41 through Figure 45) illustrate this.

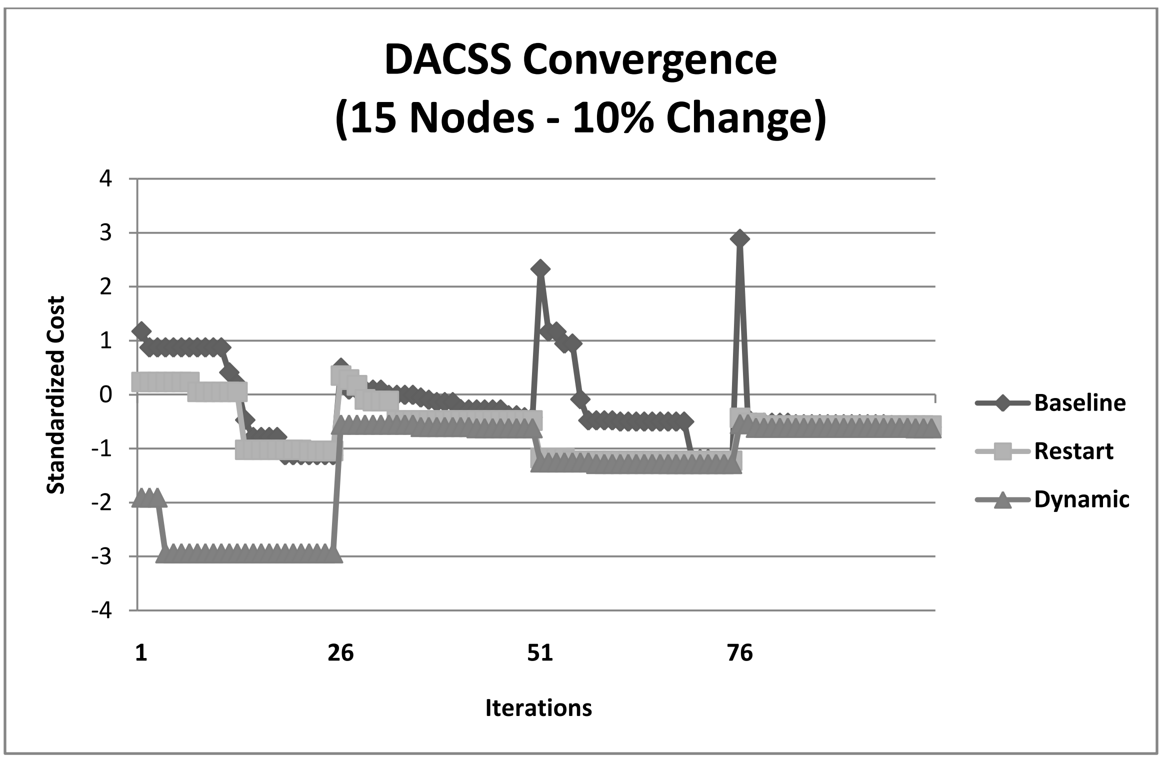


**Figure 41. DACSS Convergence - 10% Change (15 Nodes)**

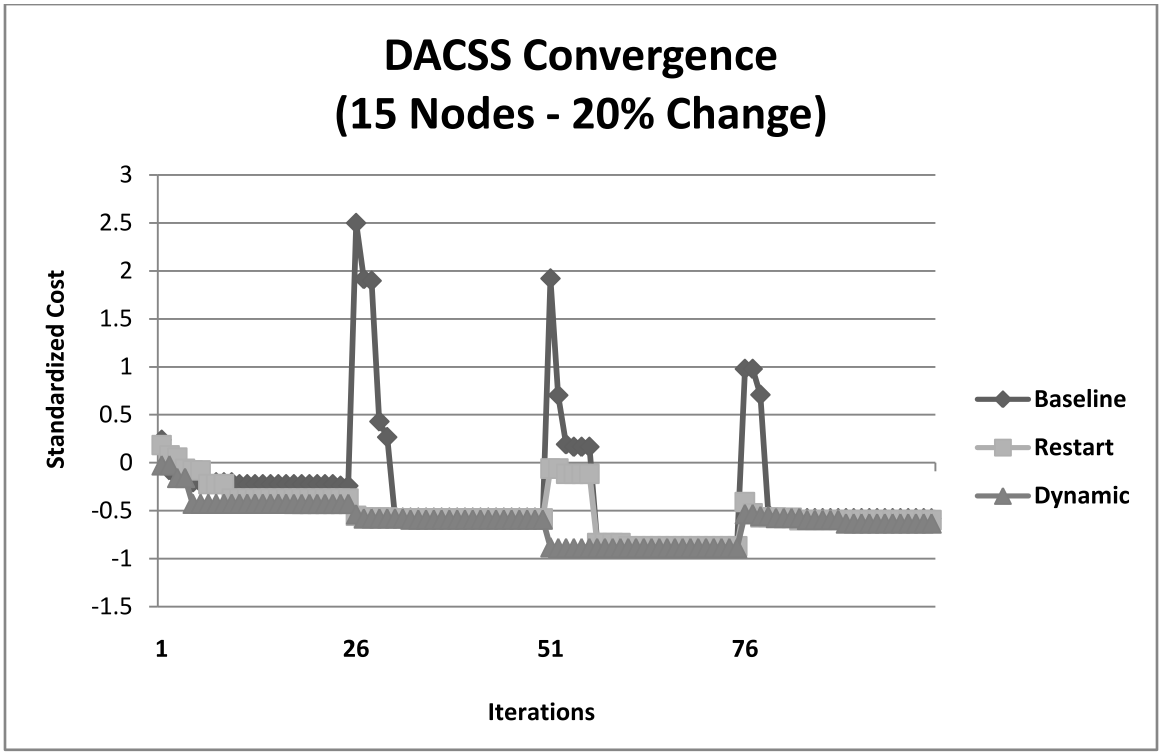


**Figure 42. DACSS Convergence - 20% Change (15 Nodes)**

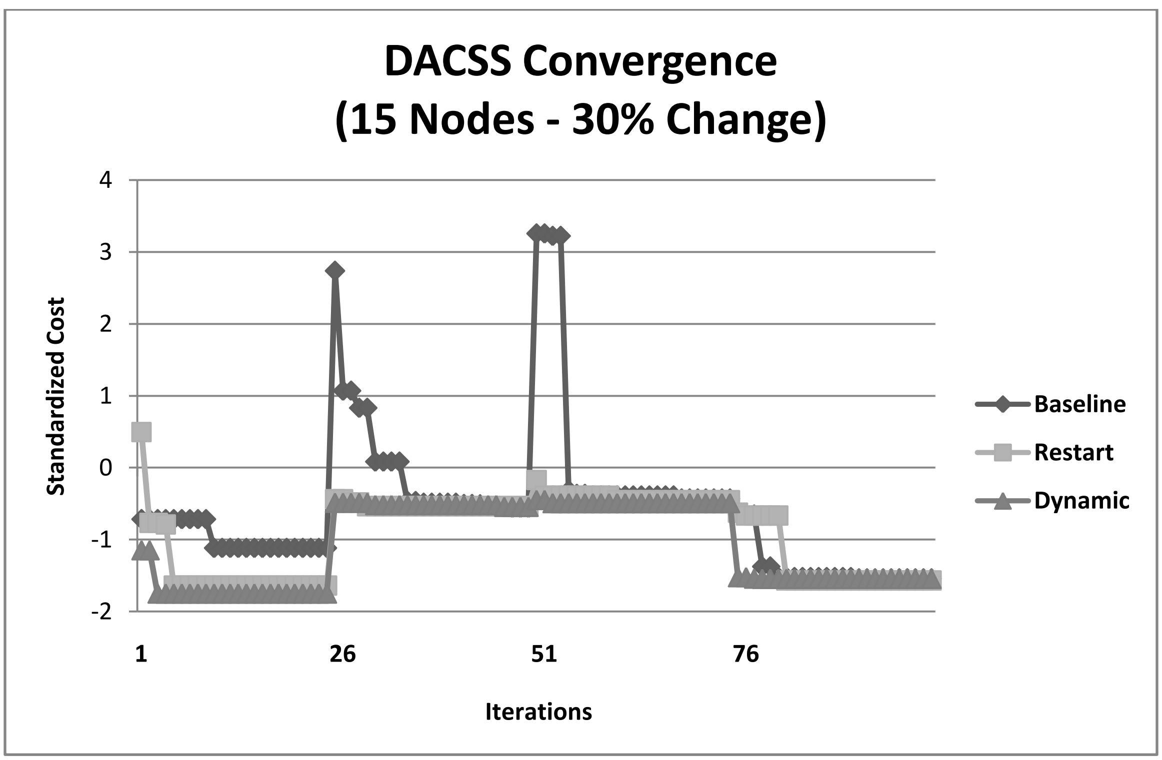


**Figure 43. DACSS Convergence - 30% Change (15 Nodes)**

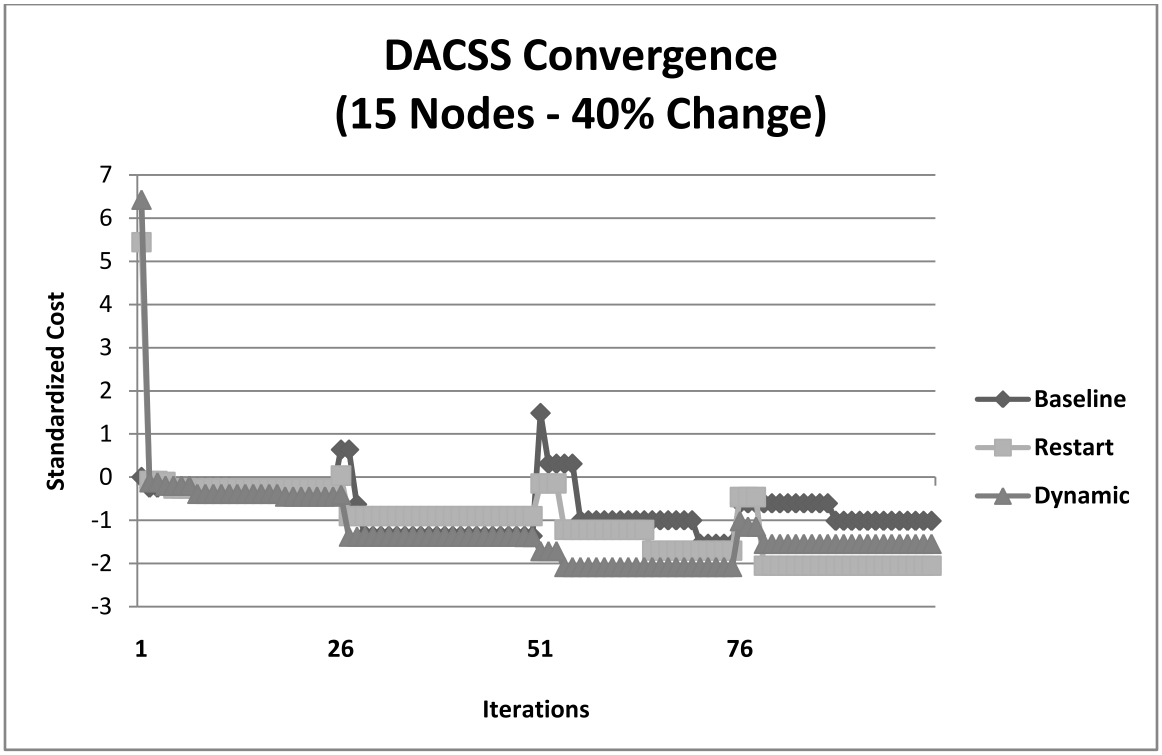


**Figure 44. DACSS Convergence - 40% Change (15 Nodes)**

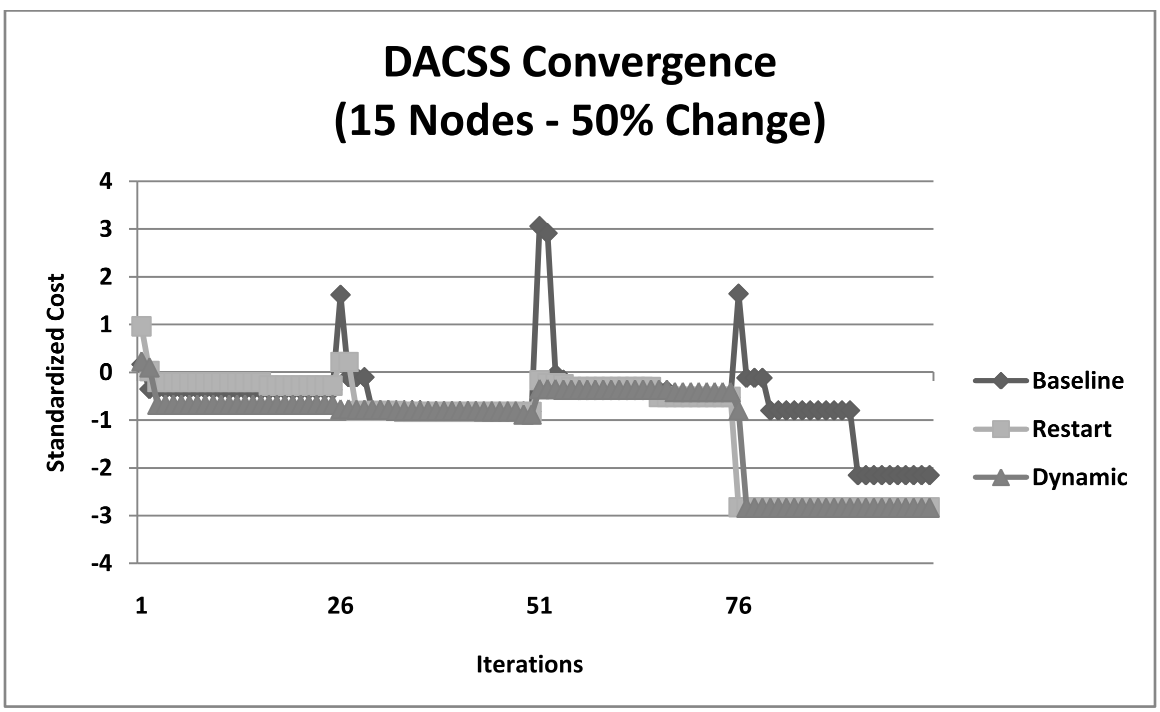


**Figure 45. DACSS Convergence - 50% Change (15 Nodes)**

The DACSS-D algorithm with respect to the 10-node and 15-node tests at all level of change (from weak to strong dynamics) is consistent with respect to convergence towards a solution. Furthermore, the charts also show the DACSS-D algorithm not only found a solution quicker, it was usually a lower cost solution as well. Most impressive is how the DACSS-D algorithm outperformed the DACSS-R algorithm which had direct knowledge of a change occurring.

## 4.5 Result Summary

The previous section provides the results of all three phases of testing. ACSS and ACSE generate lower average cost solutions than the network flow methods (Garner, 2007) for both the 10-node and 15-Node MCNDP. When compared to the MILP solutions (Erwin, 2006) the results are comparable with the 10-node MCNDP results and better than the 15-Node MCNDP solutions. The results demonstrate that the ACS algorithm is an effective approach for solving the MCNDP and DMCNDP. The following sections summarize the results from each phase.

### 4.5.1 Phase I Analysis

The ACSS is a centralized solver and uses a routing algorithm to evaluate an ant's solution. The algorithm is able to find near-optimal solutions while constructing a very reliable network topology. However, because the evaluation criteria of this approach actually routes the commodities for each ant traversal, the overall run time is considerably high. ACSS identified 10-node MCNDP solutions costing on average only

30 percent of the greedy network flow methods and just 15 percent of the average cost for the 15-node MCNDP.

For the ACSE algorithm, determining an appropriate heuristic value is necessary in order to obtain comparable results with the ACSS approach. However, using a heuristic value to estimate the actual routing cost reduces the processing time to construct a network topology solution by over 98 percent compared to ACSS and eliminates the need to route the commodities. Table 5 identifies the heuristics used to produce the best overall cost solutions for each routing strategy for a 10-node MCNDP. For the 10-node network, the overall cost solutions are slightly higher than the MILP cost solutions (Erwin, 2006), but considerably less than the network flow cost solutions (Garner, 2007). For each of these tests, ACSE dropped zero commodities and had lower overall cost solutions compared with each of the network flow approaches. The ACSE cost solutions were on average only 33 percent of the network flow solutions.

For the 15-node MCNDP, several ACSE heuristics produce very favorable results. Using a weighted value of 80 percent fixed edge cost plus 20 percent edge capacity (Test case 3) produce results with zero dropped commodities and an overall cost comparable to performing the full routing solution (ACSS). Table 6 shows the heuristics incorporated to generate the best overall cost solution for each network flow method for a 15-node MCNDP. When comparing with the network flow cost solutions, ACSE generated overall cost solutions for the 15-node MCNDP which were lower for 7 of the 8 routing strategies. The overall average ACSE cost for the 15-node MCNDP was only 65 percent of the network flow methods.

### 4.5.2 Phase 2 Analysis

Overall, the three categories responded to the dynamic change introduced to the network. The ACS algorithm and hence the ACSS and ACSE algorithms in general react to small dynamic changes appropriately. However, the ACSS-D and ACSE-D which are capable of modifying the exploration parameter converge quicker and respond better to strong dynamics in the environment. Also, the ACSS-D and ACSE-D algorithms produce lower cost solutions, on average, when compared to the ACSS-B/ACSS-R and ACSE-B/ACSE-R algorithms.

### 4.5.3 Phase 3 Analysis

The DACSS and DACSE produce comparable results with the centralized solvers, ACSS and ACSE. By using a distributed approach, decision making is removed from a central location (the network) and moved down to the lower level nodes. Instead of combining all the potential edges throughout the network and then making selections, each node runs its own ant algorithm. Each node acts independently of the other nodes and of the network as a whole. The greatest advantage to this approach is the time savings. For the DACSS 10-node tests, over 80 percent of the time the total cost means for the DACSS and ACSS were statistically the same. DACSE had an even higher rate as 92 percent were statistically equivalent to ACSE 10-node solutions. The 15-node tests produced similar results. DACSS 15-node solution costs were statistically the same as the ACSS mean costs 80 percent of the time and DACSE 15-node mean solution costs were statistically equal 95 percent of the time compared with ACSE. Nodes can now operate in parallel on a much smaller set of edges, drastically reducing the overall computation time

by 17 percent on average. The time savings from the 10-node network to the 15-node network for the ACSS algorithm decreased by 24 percent and for the ACSE algorithm by 40 percent. As this distributed approach is applied to larger networks, the time savings become even more apparent. Therefore, this approach will scale to larger networks. Also, the solution costs generated by the distributed solvers, when compared with the centralized solvers, are statistically the same over eighty percent of the time. The distributed algorithms also showed similar convergence with respect to the centralized solvers.

**4.6 Summary**

In this chapter, several approaches incorporating the ACS algorithm have shown to be effective in solving MCNDP and DMCNDP. The Phase 1 testing results show a comparison of the ACSS and ACSE approaches with previous solution methodologies, MILP and network flows. ACSS is efficient and develops lower cost solutions. ACSE introduces new heuristics which drastically reduces the run time. Phase 2 testing introduces three categories of the ACSS and ACSE methodologies and transforms the network environment from static (no change) to highly dynamic. Most noteworthy, the dynamic variation of the algorithms has the ability to modify its exploration parameter and adapt more quickly to a dynamically changing environment. Phase 3 introduces the distributed approach for both ACSS and ACSE. The *t*-test was used to determine if the results from the ACSS/DACSS and ACSE/DACSE are statistically significant. Also, the run times are drastically lower, and still produce comparable solutions in 20 percent of the time.

# V. Conclusion and Recommendations

Networks are transitioning to environments that do not rely on fixed infrastructure or preset connectivity (Joshi, Mishra, Batta, & Nagi, 2004). Networks must be mobile and capable of being rapidly deployed. A crucial characteristic of networks and especially mobile networks is topology control. The network must build and maintain a connected topology amongst the nodes and usually in a highly dynamic setting.

This research proposes three approaches for solving the Multi-commodity Capacitated Network Design Problems (MCNDPs) and the Dynamic MCNDP. The first approach implements two Ant Colony System (ACS) algorithms, Ant Colony System Standard (ACSS) and Ant Colony System Estimation (ACSE) to solve the static MCNDP with weak constraints. Then, a dynamic extension of both approaches was developed (ACSS-D and ACSE-D) and has the ability to dynamically alter its exploration parameter and automatically adjust to the dynamically changing network environment to converge on a solution quicker. Third, distributed approaches (DACSS and DACSE) were created replacing the previous centralized solvers. The distributed algorithms produce comparable results, but more importantly calculates the network topology in less than 20 percent of the computation time.

## 5.1 Application of Results

The proposed algorithms were developed to solve the static and dynamic MCNDP. ACSS and ACSE demonstrate the ability to solve the MCNDP. The solutions generated were significantly lower than previous approaches using network flow methods. On average, ACSS produced 10-node MCNDP solutions at only 30 percent of

the network flow methods and 15 percent for the 15-node MCNDP. ACSE also showed significantly lower costs (33 percent for 10-node MCNDP and 65% for 15-node MCNDP). The ACCS-Dynamic (ACSS-D) and ACSE-Dynamic (ACSE-D) found lower average cost solutions when compared with ACSS-B/ACSS-R and ACSE-B/ACSE-R. Also, ACSS-D converges quicker to a solution and responds better to change than ACSS-B and ACSS-R. The distributed solvers produce solutions which are statistically the same as the centralized solvers 80 percent of the time for DACSS and over 92 percent for DACSE. In addition, the distributed solvers drastically reduced the average computation time. DACSS for the 10-node DMCNDP computes a solution in less than 17 percent of the ACSS time; DACSE for the 10-node DMCNDP is 30 percent of the ACSE time; DACSS for the 15-node DMCNDP is less than 13 percent of the ACSS time; DACSE for the 15-node DMCNDP is 18 percent of the ACSE time. In addition, these methodologies are useful for any variation of the Network Design Problem (NDP) or any network-routing related design problems.

### 5.2 Recommendations for Future Work

The MILP solver results were included for comparison of this research with optimal solutions. However, upon further investigation the MILP solver was analyzed to verify the accuracy of its results. Although several MILP solutions generated had much lower scores than the ACSS, ACSE, DACSS, and DACSE, errors with the program were identified. Valid network topology solutions that the ant algorithms generated served as input to the MILP solver. There were instances of the MILP solver unable to find the feasible solution.

The research area deserving primary attention is the DACSE. The DACSE algorithm's evaluation function needs to be explored in greater depth. In addition, heuristic development and parameter testing needs to be further evaluated in order to ensure convergence towards an optimal solution. With the appropriate heuristics to perform the probability transition rule and evaluation function, the possibility of utilizing this approach on a much larger network becomes more feasible. Reducing the computation time enables this methodology to scale upwards and have the potential to be a viable approach.

Additionally, q-learning techniques should be explored to identify the best weights to use for the identified heuristics. This new heuristic could then be incorporated into greedy search algorithms as an additional methodology for solving the DMCNDP.

Furthermore, investigation into each node being capable of evaluating its own selection vice a global evaluation is necessary. The node would need to incorporate a new set of heuristics to evaluate its own selection beyond simply how good the choice is at the node level. The heuristic would need to capture some element of the network as a whole. Eliminating the central evaluation would reduce the dependence amongst nodes and even further extend their independence.

Also, the heuristics identified could be incorporated into other solvers to achieve even better results. This approach could be further applied to several other problem domains to include routing, assignment, scheduling, and machine learning problem types.

Lastly, sensitivity analysis with respect to the parameter settings identified in Table 4 should be explored. This could be expanded to investigate the number of generations used for the dynamic algorithm as well.

## 5.3 Conclusion

This dissertation proposes a novel approach for solving the MCNDP and DMCNDP. This research effort of incorporating ACO algorithms to solve the MCNDP and DMCNDP has contributions in several advancements: ACSS, ACSE, ACSE, Dynamic MCNDP solver, DACSS, and DACSE. First, ACSS clearly demonstrates the ant algorithms' ability to successfully solve the MCNDP. Then, ACSE explores several new heuristic components and eliminates the need to actually route commodities in order to test for a solution. This approach drastically reduces the computation time needed to construct a network topology. A second improvement is the dynamic algorithms (Category 3), which are able to quickly adapt to network changes and converge on a new solution. The last improvement is transforming the algorithms from being centralized solvers to distributed solvers. The distributed solvers produce comparable solutions with the previous centralized methodology. However, the greatest achievement is the dynamic algorithms (DACSS and DACSE) drastically reduce the required computation time to find an acceptable solution. Reducing the overall run time provides greater confidence that these approaches are scalable to much larger domain problems.

## Appendix A: Phase 1 Results - 10 Node Network

### Table 21. MILP and Network Flow Solutions (10 Nodes)

| Approach | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| LP Barrier* | 161.30 | 662.94 | N/A | 824.24 | 56.33 | 2.07 | 5.60 | 0.00 | 20.89 |
| LP Dual* | 161.30 | 662.94 | N/A | 824.24 | 56.32 | 2.07 | 5.60 | 0.00 | 17.96 |
| Combo* | 139.60 | 721.42 | N/A | 861.02 | 72.72 | 2.12 | 5.50 | 0.00 | 1.71 |
| Heuristic 1* | 144.50 | 742.83 | N/A | 887.33 | 61.08 | 2.20 | 6.00 | 0.00 | 0.48 |
| Heuristic 2* | 143.90 | 725.10 | N/A | 869.00 | 86.55 | 2.22 | 5.60 | 0.00 | 0.54 |
| Knapsack EK1** | 123.27 | 1126.07 | 2400.00 | 3649.34 | 4246.44 | 3.50 | 6.80 | 0.73 | 4.12 |
| Knapsack EK2** | 139.00 | 1107.98 | 3433.33 | 4680.32 | 5630.41 | 4.01 | 8.13 | 0.97 | 5.72 |
| Knapsack PFP1** | 176.97 | 906.42 | 0.00 | 1083.39 | 60.48 | 2.99 | 5.40 | 0.00 | 7.65 |
| KnapsackPFP2** | 190.03 | 881.22 | 500.00 | 1571.25 | 1527.75 | 2.93 | 5.60 | 0.10 | 12.71 |
| Greedy EK1** | 127.83 | 1125.89 | 2566.67 | 3820.39 | 4444.37 | 3.49 | 6.90 | 0.77 | 1.41 |
| Greedy EK2** | 139.63 | 1159.78 | 2633.33 | 3932.74 | 3891.77 | 4.07 | 7.97 | 0.80 | 1.83 |
| Greedy PFP1** | 169.50 | 928.70 | 133.33 | 1231.54 | 742.02 | 3.04 | 5.63 | 0.03 | 2.37 |
| Greedy PFP2** | 186.33 | 871.38 | 1933.33 | 2991.04 | 5983.86 | 2.93 | 5.33 | 0.43 | 4.28 |
| Greedy EK1 ACSS | 188.67 | 701.37 | 0.00 | 890.03 | 10.35 | 2.60 | 4.00 | 0.00 | 1547.16 |
| Greedy EK2 ACSS | 187.86 | 723.98 | 0.00 | 911.84 | 17.95 | 3.77 | 7.41 | 0.00 | 1926.58 |
| Greedy PFP1 ACSS | 176.73 | 693.33 | 0.00 | 870.06 | 8.11 | 2.63 | 4.00 | 0.00 | 2151.04 |
| Greedy PFP2 ACSS | 178.63 | 692.25 | 0.00 | 870.88 | 7.29 | 2.62 | 4.00 | 0.00 | 2140.26 |

*  - MILP total cost does NOT include the penalty cost. Data obtained from (Erwin, 2006).
** - Network Flow algorithms (Garner, 2007) were re-tested to include the dropped penalty cost.

### Table 22. ACSS Results (10 Nodes)

| Approach | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Greedy EK1 ACSS | 188.67 | 701.37 | 0.00 | 890.03 | 10.35 | 2.60 | 4.00 | 0.00 | 1547.16 |
| Greedy EK2 ACSS | 187.86 | 723.98 | 0.00 | 911.84 | 17.95 | 3.77 | 7.41 | 0.00 | 1926.58 |
| Greedy PFP1 ACSS | 176.73 | 693.33 | 0.00 | 870.06 | 8.11 | 2.63 | 4.00 | 0.00 | 2151.04 |
| Greedy PFP2 ACSS | 178.63 | 692.25 | 0.00 | 870.88 | 7.29 | 2.62 | 4.00 | 0.00 | 2140.26 |

**Table 23. ACSE - Test 1 (10 Nodes)**

| Fixed + Variable Comm | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 Ant | 163.87 | 770.31 | 0.00 | 934.18 | 7.41 | 2.68 | 4.00 | 0.00 | 26.29 |
| Knapsack EK2 Ant | 163.70 | 849.77 | 2366.67 | 3380.14 | 2320.70 | 3.97 | 7.97 | 0.63 | 26.94 |
| Knapsack PFP1 Ant | 163.93 | 769.04 | 0.00 | 932.98 | 9.39 | 2.68 | 4.00 | 0.00 | 28.47 |
| KnapsackPFP2 Ant | 163.87 | 767.44 | 0.00 | 931.31 | 13.71 | 2.68 | 4.00 | 0.00 | 27.82 |
| Greedy EK1 Ant | 164.00 | 769.86 | 0.00 | 933.86 | 6.04 | 2.67 | 4.00 | 0.00 | 22.55 |
| Greedy EK2 Ant | 163.63 | 852.57 | 966.67 | 1982.87 | 1644.59 | 3.96 | 7.97 | 0.27 | 23.85 |
| Greedy PFP1 Ant | 163.93 | 768.71 | 0.00 | 932.65 | 14.51 | 2.68 | 4.00 | 0.00 | 24.09 |
| Greedy PFP2 Ant | 163.87 | 769.93 | 0.00 | 933.80 | 12.81 | 2.68 | 4.00 | 0.00 | 23.79 |

**Table 24. ACSE - Test 2 (10 Nodes)**

| Fixed Only | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 Ant Fixed | 164.03 | 768.99 | 0.00 | 933.03 | 5.25 | 2.67 | 4.00 | 0.00 | 25.38 |
| Knapsack EK2 Ant Fixed | 163.77 | 840.73 | 2333.33 | 3337.84 | 2691.57 | 3.95 | 8.00 | 0.63 | 26.99 |
| Knapsack PFP1 Ant Fixed | 164.00 | 763.06 | 0.00 | 927.06 | 11.10 | 2.67 | 4.00 | 0.00 | 27.48 |
| KnapsackPFP2 Ant Fixed | 164.00 | 763.93 | 0.00 | 927.93 | 9.45 | 2.68 | 4.00 | 0.00 | 28.14 |
| Greedy EK1 Ant Fixed | 164.00 | 769.83 | 0.00 | 933.83 | 8.06 | 2.67 | 4.00 | 0.00 | 22.72 |
| Greedy EK2 Ant Fixed | 163.77 | 847.35 | 1866.67 | 2877.78 | 2400.71 | 3.96 | 8.00 | 0.53 | 23.04 |
| Greedy PFP1 Ant Fixed | 164.00 | 764.28 | 0.00 | 928.28 | 8.62 | 2.68 | 4.00 | 0.00 | 23.24 |
| Greedy PFP2 Ant Fixed | 164.00 | 764.95 | 0.00 | 928.95 | 10.29 | 2.67 | 4.00 | 0.00 | 23.28 |

## Table 25. ACSE - Test 3 (10 Nodes)

| Capacity 80/20 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time | Diff Time |
|---|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 169.00 | 815.60 | 0.00 | 984.60 | 8.34 | 2.71 | 5.00 | 0.00 | 24.22 | 24.23 |
| Knapsack EK2 | 169.00 | 946.31 | 0.00 | 1115.31 | 3.96 | 3.98 | 8.00 | 0.00 | 25.72 | 25.73 |
| Knapsack PFP1 | 169.03 | 821.00 | 0.00 | 990.03 | 9.70 | 2.73 | 5.00 | 0.00 | 25.39 | 25.47 |
| KnapsackPFP2 | 169.03 | 846.49 | 0.00 | 1015.53 | 4.71 | 2.75 | 4.97 | 0.00 | 26.34 | 26.33 |
| Greedy EK1 | 169.00 | 813.19 | 0.00 | 982.19 | 6.36 | 2.71 | 5.00 | 0.00 | 21.42 | 22.23 |
| Greedy EK2 | 169.03 | 945.98 | 0.00 | 1115.01 | 17.97 | 3.99 | 8.00 | 0.00 | 21.80 | 22.40 |
| Greedy PFP1 | 169.00 | 832.61 | 0.00 | 1001.61 | 10.86 | 2.72 | 5.00 | 0.00 | 22.41 | 23.60 |
| Greedy PFP2 | 169.07 | 830.69 | 0.00 | 999.76 | 13.16 | 2.72 | 4.97 | 0.00 | 22.03 | 23.03 |

## Table 26. ACSE - Test 4 (10 Nodes)

| Capacity 90/10 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time | Diff Time |
|---|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 167.07 | 775.77 | 0.00 | 942.84 | 12.71 | 2.68 | 4.03 | 0.00 | 24.17 | 24.23 |
| Knapsack EK2 | 167.07 | 860.34 | 0.00 | 1027.41 | 20.83 | 3.84 | 8.00 | 0.00 | 25.01 | 25.07 |
| Knapsack PFP1 | 167.07 | 788.41 | 0.00 | 955.48 | 9.26 | 2.68 | 4.03 | 0.00 | 27.88 | 27.97 |
| KnapsackPFP2 | 166.97 | 784.47 | 0.00 | 951.43 | 8.56 | 2.69 | 4.00 | 0.00 | 26.82 | 26.80 |
| Greedy EK1 | 167.03 | 776.49 | 0.00 | 943.52 | 13.76 | 2.68 | 4.03 | 0.00 | 21.29 | 21.73 |
| Greedy EK2 | 167.07 | 853.20 | 333.33 | 1353.60 | 1024.73 | 3.84 | 7.67 | 0.10 | 21.68 | 22.30 |
| Greedy PFP1 | 167.17 | 791.46 | 0.00 | 958.62 | 15.51 | 2.69 | 4.07 | 0.00 | 22.24 | 22.83 |
| Greedy PFP2 | 166.97 | 787.72 | 0.00 | 954.68 | 10.50 | 2.69 | 4.00 | 0.00 | 21.83 | 22.37 |

## Table 27. ACSE - Test 5 (10 Nodes)

| Capacity 95/05 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time | Diff Time |
|---|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 164.00 | 768.80 | 0.00 | 932.80 | 2.44 | 2.68 | 4.00 | 0.00 | 24.52 | 24.60 |
| Knapsack EK2 | 164.00 | 832.07 | 2000.00 | 2996.07 | 2019.17 | 3.92 | 8.00 | 1.00 | 26.27 | 26.30 |
| Knapsack PFP1 | 164.00 | 761.68 | 0.00 | 925.68 | 4.55 | 2.69 | 4.00 | 0.00 | 27.58 | 27.63 |
| Knapsack PFP2 | 164.07 | 776.87 | 0.00 | 940.93 | 14.71 | 2.69 | 4.00 | 0.00 | 25.76 | 25.77 |
| Greedy EK1 | 164.00 | 767.74 | 0.00 | 931.74 | 7.46 | 2.67 | 4.00 | 0.00 | 21.90 | 23.03 |
| Greedy EK2 | 163.77 | 847.33 | 2200.00 | 3211.09 | 2611.39 | 3.96 | 8.00 | 0.60 | 21.36 | 21.60 |
| Greedy PFP1 | 164.00 | 764.18 | 0.00 | 928.18 | 10.02 | 2.68 | 4.00 | 0.00 | 21.86 | 22.17 |
| Greedy PFP2 | 164.00 | 771.26 | 0.00 | 935.26 | 13.35 | 2.68 | 4.00 | 0.00 | 22.04 | 22.87 |

## Table 28. ACSE - Test 6 (10 Nodes)

| Capacity 80/20 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 167.13 | 775.44 | 0.00 | 942.57 | 14.56 | 2.69 | 4.10 | 0.00 | 27.67 |
| Knapsack EK2 | 167.03 | 851.06 | 1066.67 | 2084.76 | 1783.28 | 3.86 | 7.50 | 0.27 | 27.38 |
| Knapsack PFP1 | 166.97 | 773.62 | 0.00 | 940.59 | 16.46 | 2.68 | 4.10 | 0.00 | 28.30 |
| KnapsackPFP2 | 166.87 | 794.23 | 0.00 | 961.09 | 11.18 | 2.69 | 4.10 | 0.00 | 28.87 |
| Greedy EK1 | 167.27 | 776.73 | 0.00 | 944.00 | 13.82 | 2.68 | 4.10 | 0.00 | 23.69 |
| Greedy EK2 | 167.37 | 864.48 | 2333.33 | 3365.18 | 1587.48 | 3.90 | 7.73 | 0.77 | 25.37 |
| Greedy PFP1 | 166.97 | 779.44 | 0.00 | 946.40 | 17.46 | 2.69 | 4.17 | 0.00 | 24.10 |
| Greedy PFP2 | 167.17 | 778.24 | 0.00 | 945.41 | 24.15 | 2.69 | 4.13 | 0.00 | 25.47 |

## Table 29. ACSE - Test 7 (10 Nodes)

| Capacity 90/10 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 166.63 | 770.17 | 0.00 | 936.80 | 7.06 | 2.68 | 4.00 | 0.00 | 26.30 |
| Knapsack EK2 | 166.30 | 859.36 | 933.33 | 1959.00 | 1713.80 | 3.89 | 7.97 | 0.23 | 29.51 |
| Knapsack PFP1 | 166.53 | 786.88 | 0.00 | 953.41 | 10.43 | 2.68 | 4.00 | 0.00 | 27.95 |
| KnapsackPFP2 | 166.40 | 794.88 | 0.00 | 961.28 | 9.62 | 2.70 | 4.00 | 0.00 | 30.28 |
| Greedy EK1 | 166.53 | 769.78 | 0.00 | 936.31 | 8.40 | 2.68 | 4.03 | 0.00 | 24.62 |
| Greedy EK2 | 165.57 | 858.07 | 2100.00 | 3123.64 | 2621.19 | 3.94 | 8.00 | 0.70 | 25.90 |
| Greedy PFP1 | 166.60 | 785.30 | 0.00 | 951.90 | 5.98 | 2.70 | 4.00 | 0.00 | 24.26 |
| Greedy PFP2 | 166.40 | 782.83 | 0.00 | 949.23 | 10.64 | 2.70 | 4.03 | 0.00 | 24.36 |

## Table 30. ACSE - Test 8 (10 Nodes)

| Capacity 95/05 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 164.00 | 772.20 | 0.00 | 936.20 | 5.90 | 2.68 | 4.00 | 0.00 | 24.56 |
| Knapsack EK2 | 164.00 | 842.40 | 5000.00 | 6006.40 | 2038.26 | 3.98 | 8.00 | 1.50 | 30.36 |
| Knapsack PFP1 | 164.00 | 764.48 | 0.00 | 928.48 | 3.79 | 2.68 | 4.00 | 0.00 | 30.06 |
| Knapsack PFP2 | 164.00 | 773.83 | 0.00 | 937.83 | 2.75 | 2.69 | 4.00 | 0.00 | 29.99 |
| Greedy EK1 | 164.00 | 771.20 | 0.00 | 935.20 | 0.00 | 2.68 | 4.00 | 0.00 | 25.18 |
| Greedy EK2 | 164.00 | 870.85 | 0.00 | 1034.85 | 0.00 | 4.02 | 8.00 | 0.00 | 25.63 |
| Greedy PFP1 | 164.00 | 783.00 | 0.00 | 947.00 | 0.00 | 2.68 | 4.00 | 0.00 | 25.26 |
| Greedy PFP2 | 164.00 | 780.00 | 0.00 | 944.00 | 0.00 | 2.68 | 4.00 | 0.00 | 24.22 |

## Table 31. ACSE - Test 9 (10 Nodes)

| Value 80/20 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 167.50 | 767.37 | 0.00 | 934.87 | 8.44 | 2.68 | 4.00 | 0.00 | 25.53 |
| Knapsack EK2 | 169.00 | 915.43 | 4500.00 | 5584.45 | 4556.13 | 3.97 | 8.00 | 1.00 | 26.89 |
| Knapsack PFP1 | 168.00 | 817.13 | 0.00 | 985.13 | 19.35 | 2.73 | 4.50 | 0.00 | 27.72 |
| KnapsackPFP2 | 168.23 | 797.24 | 0.00 | 965.47 | 21.90 | 2.71 | 4.00 | 0.00 | 26.69 |
| Greedy EK1 | 168.17 | 796.58 | 0.00 | 964.74 | 21.61 | 2.69 | 4.20 | 0.00 | 22.28 |
| Greedy EK2 | 167.97 | 866.92 | 1700.00 | 2734.89 | 2869.85 | 3.93 | 7.73 | 0.40 | 23.20 |
| Greedy PFP1 | 168.90 | 793.73 | 0.00 | 962.63 | 27.56 | 2.71 | 4.47 | 0.00 | 23.27 |
| Greedy PFP2 | 168.77 | 793.83 | 0.00 | 962.60 | 33.21 | 2.71 | 4.30 | 0.00 | 22.66 |

## Table 32. ACSE - Test 10 (10 Nodes)

| Value 90/10 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 165.57 | 761.67 | 0.00 | 927.23 | 8.34 | 2.66 | 4.00 | 0.00 | 26.16 |
| Knapsack EK2 | 166.20 | 854.28 | 200.00 | 1220.48 | 750.81 | 3.91 | 8.00 | 0.07 | 25.60 |
| Knapsack PFP1 | 166.50 | 776.40 | 0.00 | 942.90 | 1.53 | 2.69 | 4.00 | 0.00 | 28.56 |
| KnapsackPFP2 | 165.50 | 780.40 | 0.00 | 945.90 | 21.26 | 2.68 | 4.00 | 0.00 | 27.97 |
| Greedy EK1 | 166.30 | 789.84 | 0.00 | 956.14 | 25.42 | 2.69 | 4.27 | 0.00 | 22.67 |
| Greedy EK2 | 166.03 | 858.11 | 1366.67 | 2390.81 | 2281.17 | 3.92 | 7.53 | 0.37 | 23.23 |
| Greedy PFP1 | 166.43 | 789.67 | 0.00 | 956.10 | 30.38 | 2.70 | 4.23 | 0.00 | 23.35 |
| Greedy PFP2 | 165.93 | 782.29 | 0.00 | 948.22 | 19.85 | 2.70 | 4.10 | 0.00 | 23.78 |

## Table 33. ACSE - Test 11 (10 Nodes)

| Value 95/05 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 164.00 | 776.80 | 0.00 | 940.80 | 1.83 | 2.67 | 4.00 | 0.00 | 23.36 |
| Knapsack EK2 | 164.00 | 853.40 | 1500.00 | 2517.40 | 1503.47 | 3.92 | 8.00 | 0.50 | 27.85 |
| Knapsack PFP1 | 165.00 | 782.60 | 0.00 | 947.60 | 8.54 | 2.69 | 4.00 | 0.00 | 26.73 |
| Knapsack PFP2 | 165.50 | 788.13 | 0.00 | 953.63 | 3.02 | 2.69 | 4.00 | 0.00 | 25.75 |
| Greedy EK1 | 164.63 | 774.09 | 0.00 | 938.73 | 12.20 | 2.68 | 4.00 | 0.00 | 21.34 |
| Greedy EK2 | 164.77 | 852.62 | 1100.00 | 2117.39 | 1724.24 | 3.93 | 7.73 | 0.30 | 22.39 |
| Greedy PFP1 | 164.73 | 772.58 | 0.00 | 937.31 | 12.53 | 2.68 | 4.00 | 0.00 | 22.04 |
| Greedy PFP2 | 164.63 | 773.57 | 0.00 | 938.21 | 14.08 | 2.68 | 4.03 | 0.00 | 22.03 |

## Table 34. ACSE - Test 12 (10 Nodes)

| Value 80/20 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 167.40 | 793.61 | 0.00 | 961.01 | 27.85 | 2.70 | 4.33 | 0.00 | 24.33 |
| Knapsack EK2 | 167.40 | 857.95 | 800.00 | 1825.36 | 1621.72 | 3.96 | 7.40 | 0.20 | 26.45 |
| Knapsack PFP1 | 167.27 | 798.51 | 0.00 | 965.77 | 18.38 | 2.71 | 4.50 | 0.00 | 26.86 |
| KnapsackPFP2 | 167.90 | 786.84 | 0.00 | 954.74 | 24.15 | 2.69 | 4.33 | 0.00 | 27.84 |
| Greedy EK1 | 165.20 | 777.20 | 0.00 | 942.40 | 3.25 | 2.67 | 4.00 | 0.00 | 23.09 |
| Greedy EK2 | 165.30 | 846.76 | 2466.67 | 3478.73 | 3327.25 | 3.92 | 7.67 | 0.70 | 22.67 |
| Greedy PFP1 | 165.37 | 772.53 | 0.00 | 937.89 | 5.92 | 2.67 | 4.00 | 0.00 | 23.86 |
| Greedy PFP2 | 165.40 | 774.52 | 0.00 | 939.92 | 4.09 | 2.67 | 4.00 | 0.00 | 24.06 |

## Table 35. ACSE - Test 13 (10 Nodes)

| Value 90/10 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 166.67 | 789.89 | 0.00 | 956.56 | 15.40 | 2.69 | 4.07 | 0.00 | 24.91 |
| Knapsack EK2 | 166.00 | 863.46 | 3200.00 | 4229.46 | 3419.09 | 3.93 | 7.53 | 0.73 | 24.60 |
| Knapsack PFP1 | 166.60 | 785.76 | 0.00 | 952.36 | 20.47 | 2.68 | 4.23 | 0.00 | 27.84 |
| KnapsackPFP2 | 166.90 | 798.97 | 0.00 | 965.87 | 25.65 | 2.70 | 4.50 | 0.00 | 29.32 |
| Greedy EK1 | 167.13 | 799.42 | 0.00 | 966.55 | 13.53 | 2.68 | 4.10 | 0.00 | 25.67 |
| Greedy EK2 | 167.20 | 872.19 | 466.67 | 1506.06 | 2547.90 | 3.93 | 7.60 | 0.10 | 25.06 |
| Greedy PFP1 | 167.63 | 774.96 | 0.00 | 942.59 | 9.43 | 2.68 | 4.10 | 0.00 | 24.86 |
| Greedy PFP2 | 167.13 | 773.63 | 0.00 | 940.77 | 10.67 | 2.69 | 4.10 | 0.00 | 23.51 |

## Table 36. ACSE - Test 14 (10 Nodes)

| Value95/05 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 164.00 | 773.00 | 0.00 | 937.00 | 6.10 | 2.68 | 4.00 | 0.00 | 26.04 |
| Knapsack EK2 | 164.47 | 872.16 | 133.33 | 1169.96 | 723.32 | 3.95 | 8.00 | 0.03 | 27.70 |
| Knapsack PFP1 | 164.40 | 762.49 | 0.00 | 926.89 | 6.38 | 2.68 | 4.00 | 0.00 | 27.15 |
| Knapsack PFP2 | 165.73 | 772.63 | 0.00 | 938.36 | 6.13 | 2.68 | 4.00 | 0.00 | 28.36 |
| Greedy EK1 | 165.07 | 766.40 | 0.00 | 931.47 | 5.58 | 2.67 | 4.00 | 0.00 | 24.89 |
| Greedy EK2 | 164.93 | 856.52 | 0.00 | 1021.45 | 5.28 | 3.96 | 7.53 | 0.00 | 25.47 |
| Greedy PFP1 | 164.87 | 755.75 | 0.00 | 920.61 | 3.63 | 2.69 | 4.00 | 0.00 | 25.00 |
| Greedy PFP2 | 164.73 | 756.36 | 0.00 | 921.09 | 3.53 | 2.69 | 4.00 | 0.00 | 25.56 |

## Appendix B: Phase 1 Results - 15 Node Network

**Table 37. MILP and Network Flow Solutions (15 Nodes)**

| Approach | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| LP Barrier* | 302.20 | 1579.01 | N/A | 1881.21 | 120.00 | 2.38 | 7.30 | 5.80 | 707.27 |
| LP Dual* | 302.00 | 1616.31 | N/A | 1918.31 | 161.64 | 2.41 | 7.50 | 5.70 | 762.57 |
| LP Primal* | 291.80 | 1638.60 | N/A | 1930.40 | 175.92 | 2.42 | 7.10 | 5.80 | 867.87 |
| Combo* | 264.50 | 1602.55 | N/A | 1867.05 | 117.97 | 2.40 | 7.40 | 8.60 | 142.94 |
| Heuristic 1* | 245.30 | 1703.92 | N/A | 1949.22 | 176.04 | 2.55 | 8.20 | 16.80 | 3.55 |
| Heuristic 2* | 250.30 | 1640.68 | N/A | 1890.98 | 151,38 | 2.51 | 7.80 | 19.00 | 3.64 |
| Knapsack EK1** | 264.20 | 3195.14 | 6666.67 | 10126.01 | 8079.66 | 4.01 | 9.50 | 1.70 | 45.03 |
| Knapsack EK2** | 281.43 | 3033.35 | 13533.33 | 16848.12 | 11913.31 | 4.73 | 10.70 | 3.77 | 64.34 |
| Knapsack PFP1** | 321.93 | 2400.28 | 3266.67 | 5988.88 | 7298.50 | 3.28 | 6.80 | 0.83 | 77.38 |
| KnapsackPFP2** | 334.83 | 2240.27 | 21066.67 | 23641.75 | 22157.38 | 3.18 | 6.77 | 6.33 | 197.77 |
| Greedy EK1** | 252.33 | 3208.69 | 5800.00 | 9261.03 | 9095.95 | 4.06 | 9.53 | 1.43 | 13.21 |
| Greedy EK2** | 270.77 | 3016.27 | 16933.33 | 20220.38 | 11768.29 | 4.72 | 10.80 | 4.70 | 19.20 |
| Greedy PFP1** | 318.83 | 2490.44 | 2766.67 | 5575.94 | 6628.40 | 3.37 | 7.33 | 0.80 | 23.43 |
| Greedy PFP2** | 327.27 | 2256.37 | 16866.67 | 19450.31 | 15486.60 | 3.17 | 6.57 | 4.93 | 49.24 |

*   - MILP total cost does NOT include the penalty cost. Data obtained from (Erwin, 2006).
** - Network Flow algorithms (Garner, 2007) were re-tested to include the dropped penalty cost.

**Table 38. ACSS (15 Nodes)**

| Approach | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Greedy EK1 ACSS | 282.57 | 1784.18 | 0.00 | 2066.75 | 22.90 | 2.79 | 4.13 | 0.00 | 8380.81 |
| Greedy EK2 ACSS | 285.00 | 1787.72 | 0.00 | 2072.72 | 0.00 | 4.30 | 9.78 | 0.00 | 11439.74 |
| Greedy PFP1 ACSS | 277.97 | 1843.54 | 0.00 | 2121.50 | 29.41 | 2.83 | 4.57 | 0.00 | 12625.34 |
| Greedy PFP2 ACSS | 277.71 | 1867.65 | 0.00 | 2145.36 | 39.54 | 2.84 | 4.71 | 0.00 | 12386.57 |

**Table 39. ACSE - Test 1 (15 Nodes)**

| Fixed + Variable Comm | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 Ant | 247.20 | 1914.79 | 14200.00 | 16361.99 | 5048.36 | 2.88 | 5.00 | 3.60 | 210.17 |
| Knapsack EK2 Ant | 247.37 | 1959.63 | 22033.33 | 24240.32 | 6300.42 | 4.46 | 10.73 | 5.90 | 245.09 |
| Knapsack PFP1 Ant | 265.87 | 2015.40 | 3866.67 | 6147.93 | 3757.12 | 2.92 | 5.00 | 0.97 | 235.75 |
| KnapsackPFP2 Ant | 258.60 | 1998.93 | 9766.67 | 12024.18 | 4295.02 | 2.92 | 5.00 | 2.77 | 227.56 |
| Greedy EK1 Ant | 247.20 | 1924.53 | 16566.67 | 18738.40 | 936.13 | 2.89 | 4.97 | 4.20 | 169.95 |
| Greedy EK2 Ant | 246.53 | 1954.67 | 21933.33 | 24134.54 | 0.00 | 4.44 | 10.67 | 5.70 | 160.50 |
| Greedy PFP1 Ant | 266.23 | 2007.13 | 3466.67 | 5740.03 | 3307.22 | 2.92 | 5.00 | 0.97 | 164.01 |
| Greedy PFP2 Ant | 257.60 | 1998.64 | 7633.33 | 9889.58 | 4652.11 | 2.92 | 5.00 | 2.17 | 168.47 |

**Table 40. ACSE - Test 2 (15 Nodes)**

| Fixed Only | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 Ant | 248.43 | 1919.69 | 7900.00 | 10068.12 | 6567.36 | 2.87 | 4.80 | 2.07 | 213.35 |
| Knapsack EK2 Ant | 248.23 | 1942.88 | 14333.33 | 16524.45 | 7748.09 | 4.43 | 10.17 | 4.00 | 222.26 |
| Knapsack PFP1 Ant | 253.17 | 1979.27 | 4333.33 | 6565.77 | 2271.55 | 2.91 | 5.00 | 1.13 | 221.12 |
| KnapsackPFP2 Ant | 252.27 | 1982.57 | 6766.67 | 9001.50 | 3602.79 | 2.91 | 5.00 | 1.83 | 218.18 |
| Greedy EK1 Ant | 248.67 | 1924.82 | 6266.67 | 8440.15 | 5218.43 | 2.87 | 4.90 | 1.60 | 162.36 |
| Greedy EK2 Ant | 248.73 | 1950.03 | 13466.67 | 15665.44 | 7675.91 | 4.44 | 10.33 | 3.67 | 167.34 |
| Greedy PFP1 Ant | 255.37 | 1987.57 | 3666.67 | 5909.60 | 2677.55 | 2.91 | 5.00 | 0.90 | 175.14 |
| Greedy PFP2 Ant | 252.43 | 1986.24 | 6866.67 | 9105.35 | 3940.50 | 2.91 | 5.00 | 1.80 | 221.63 |

**Table 41. ACSE - Test 3 (15 Nodes)**

| Capacity 80/20 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 261.00 | 1891.82 | 15500.00 | 17652.81 | 508.57 | 2.87 | 5.00 | 4.00 | 200.58 |
| Knapsack EK2 | 261.60 | 1957.27 | 22833.33 | 25052.22 | 4340.71 | 4.55 | 10.03 | 5.70 | 221.69 |
| Knapsack PFP1 | 275.30 | 2101.84 | 0.00 | 2377.14 | 16.38 | 2.91 | 4.57 | 0.00 | 203.92 |
| KnapsackPFP2 | 277.57 | 2033.15 | 7466.67 | 9777.40 | 1530.62 | 2.87 | 5.00 | 2.30 | 244.14 |
| Greedy EK1 | 261.37 | 1911.72 | 16166.67 | 18339.75 | 3149.68 | 2.87 | 4.87 | 4.23 | 148.66 |
| Greedy EK2 | 261.90 | 1965.18 | 25733.33 | 27960.42 | 4008.57 | 4.57 | 10.80 | 6.90 | 170.88 |
| Greedy PFP1 | 277.10 | 2066.47 | 3066.67 | 5410.24 | 2506.96 | 2.89 | 4.97 | 0.80 | 186.89 |
| Greedy PFP2 | 272.20 | 2054.11 | 9166.67 | 11492.98 | 4523.98 | 2.89 | 4.90 | 2.30 | 172.36 |

**Table 42. ACSE - Test 4 (15 Nodes)**

| Capacity 90/10 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 250.70 | 1914.75 | 14100.00 | 16265.45 | 2337.02 | 2.88 | 5.00 | 3.90 | 194.62 |
| Knapsack EK2 | 250.53 | 1952.98 | 23366.67 | 25570.17 | 2772.34 | 4.52 | 10.83 | 5.63 | 241.01 |
| Knapsack PFP1 | 257.10 | 2040.86 | 2000.00 | 4297.96 | 1990.16 | 2.91 | 4.93 | 0.50 | 213.41 |
| KnapsackPFP2 | 272.67 | 2032.56 | 3266.67 | 5571.89 | 861.53 | 2.90 | 5.03 | 1.10 | 216.87 |
| Greedy EK1 | 250.60 | 1906.55 | 16600.00 | 18757.14 | 1277.69 | 2.88 | 4.77 | 4.17 | 161.03 |
| Greedy EK2 | 250.73 | 1965.05 | 23900.00 | 26115.79 | 2631.59 | 4.55 | 10.80 | 6.20 | 170.71 |
| Greedy PFP1 | 268.53 | 2056.01 | 1766.67 | 4091.21 | 2004.07 | 2.90 | 5.00 | 0.53 | 195.59 |
| Greedy PFP2 | 261.07 | 2037.29 | 8433.33 | 10731.69 | 4485.03 | 2.91 | 5.00 | 2.13 | 187.40 |

**Table 43. ACSE - Test 5 (15 Nodes)**

| Capacity 95/05 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 246.47 | 1916.89 | 15066.67 | 17230.03 | 4424.22 | 2.88 | 4.83 | 4.20 | 196.89 |
| Knapsack EK2 | 246.37 | 1951.88 | 21833.33 | 24031.58 | 3461.19 | 4.50 | 10.70 | 5.57 | 226.27 |
| Knapsack PFP1 | 254.73 | 2046.55 | 5000.00 | 7301.29 | 29.46 | 2.92 | 5.00 | 1.00 | 231.84 |
| Knapsack PFP2 | 258.13 | 1966.08 | 8400.00 | 10624.22 | 5428.52 | 2.91 | 5.00 | 2.43 | 235.89 |
| Greedy EK1 | 246.17 | 1915.91 | 16466.67 | 18628.75 | 1357.34 | 2.88 | 4.83 | 4.13 | 157.26 |
| Greedy EK2 | 246.33 | 1937.29 | 24300.00 | 26483.63 | 4172.00 | 4.47 | 10.57 | 6.20 | 161.48 |
| Greedy PFP1 | 260.07 | 1993.33 | 3433.33 | 5686.73 | 3836.60 | 2.90 | 4.90 | 0.93 | 187.48 |
| Greedy PFP2 | 260.90 | 2001.97 | 8333.33 | 10596.20 | 4458.66 | 2.91 | 5.00 | 2.13 | 191.75 |

**Table 44. ACSE - Test 6 (15 Nodes)**

| Capacity 80/20 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 249.40 | 1909.66 | 13666.67 | 15825.73 | 6227.67 | 2.87 | 4.40 | 3.00 | 219.38 |
| Knapsack EK2 | 249.37 | 1981.48 | 17266.67 | 19497.51 | 9049.82 | 4.53 | 10.77 | 4.70 | 247.15 |
| Knapsack PFP1 | 258.70 | 2006.09 | 4333.33 | 6598.12 | 1877.41 | 2.91 | 5.00 | 1.60 | 232.43 |
| KnapsackPFP2 | 262.97 | 2022.50 | 6433.33 | 8718.80 | 5215.28 | 2.92 | 5.00 | 1.77 | 261.57 |
| Greedy EK1 | 249.87 | 1909.11 | 12400.00 | 14558.99 | 5588.71 | 2.86 | 5.00 | 3.10 | 155.95 |
| Greedy EK2 | 248.27 | 1913.27 | 22566.67 | 24728.19 | 6224.41 | 4.43 | 10.77 | 5.67 | 161.01 |
| Greedy PFP1 | 269.60 | 2032.71 | 1000.00 | 3302.31 | 1843.65 | 2.91 | 5.00 | 0.23 | 161.65 |
| Greedy PFP2 | 257.43 | 2014.97 | 5933.33 | 8205.74 | 2586.56 | 2.90 | 5.00 | 1.47 | 167.23 |

**Table 45. ACSE - Test 7 (15 Nodes)**

| Capacity 90/10 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 248.97 | 1934.84 | 14300.00 | 16483.81 | 6794.05 | 2.87 | 5.00 | 3.10 | 221.47 |
| Knapsack EK2 | 248.77 | 1909.28 | 19933.33 | 22091.38 | 6859.02 | 4.47 | 10.60 | 6.33 | 263.19 |
| Knapsack PFP1 | 257.20 | 2038.28 | 3466.67 | 5762.15 | 1718.97 | 2.93 | 5.00 | 0.87 | 226.98 |
| KnapsackPFP2 | 252.03 | 2010.36 | 8666.67 | 10929.06 | 4834.32 | 2.91 | 5.00 | 2.17 | 231.17 |
| Greedy EK1 | 248.87 | 1956.86 | 11366.67 | 13572.39 | 6529.32 | 2.88 | 4.17 | 2.70 | 154.50 |
| Greedy EK2 | 248.60 | 1952.82 | 16200.00 | 18401.42 | 6849.19 | 4.48 | 10.67 | 4.67 | 161.86 |
| Greedy PFP1 | 255.97 | 2017.98 | 1733.33 | 4007.28 | 2008.41 | 2.91 | 4.80 | 0.43 | 162.44 |
| Greedy PFP2 | 257.17 | 2007.19 | 6500.00 | 8764.35 | 2558.04 | 2.92 | 4.47 | 1.50 | 166.03 |

**Table 46. ACSE - Test 8 (15 Nodes)**

| Capacity 95/05 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 248.37 | 1909.91 | 9033.33 | 11191.60 | 6451.24 | 2.86 | 4.73 | 2.27 | 218.15 |
| Knapsack EK2 | 248.80 | 1921.30 | 16966.67 | 19136.77 | 8287.38 | 4.42 | 10.07 | 4.83 | 241.14 |
| Knapsack PFP1 | 258.23 | 1983.42 | 2366.67 | 4608.32 | 1728.26 | 2.91 | 5.00 | 0.67 | 230.15 |
| Knapsack PFP2 | 259.23 | 1958.09 | 4000.00 | 6217.33 | 32.38 | 2.90 | 5.00 | 1.00 | 238.53 |
| Greedy EK1 | 248.07 | 1894.32 | 8200.00 | 10342.38 | 6038.01 | 2.86 | 4.23 | 2.30 | 155.65 |
| Greedy EK2 | 247.97 | 1979.90 | 19700.00 | 21927.86 | 8871.58 | 4.51 | 10.70 | 4.83 | 161.92 |
| Greedy PFP1 | 251.27 | 2019.34 | 3433.33 | 5703.94 | 2285.13 | 2.91 | 5.00 | 0.70 | 162.48 |
| Greedy PFP2 | 255.80 | 2004.20 | 6900.00 | 9160.00 | 2663.39 | 2.91 | 5.00 | 1.70 | 166.63 |

## Table 47. ACSE - Test 9 (15 Nodes)

| Value 80/20 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 250.67 | 1924.35 | 15500.00 | 17675.00 | 540.70 | 2.88 | 5.00 | 4.00 | 204.35 |
| Knapsack EK2 | 257.73 | 2006.71 | 22800.00 | 25064.44 | 5727.47 | 4.55 | 10.00 | 6.13 | 226.46 |
| Knapsack PFP1 | 266.93 | 2069.42 | 3166.67 | 5503.02 | 2323.24 | 2.93 | 5.40 | 0.67 | 209.90 |
| KnapsackPFP2 | 270.80 | 2000.43 | 14333.33 | 16604.58 | 2506.98 | 2.92 | 5.00 | 3.47 | 236.04 |
| Greedy EK1 | 255.83 | 1953.36 | 16200.00 | 18409.19 | 2672.09 | 2.89 | 5.03 | 4.13 | 154.93 |
| Greedy EK2 | 256.27 | 2032.22 | 24666.67 | 26955.16 | 9822.45 | 4.58 | 10.47 | 6.53 | 162.40 |
| Greedy PFP1 | 272.93 | 2017.17 | 1733.33 | 4023.44 | 2795.77 | 2.91 | 5.03 | 0.43 | 187.67 |
| Greedy PFP2 | 271.67 | 2007.38 | 7266.67 | 9545.71 | 4004.81 | 2.91 | 5.00 | 1.97 | 192.39 |

## Table 48. ACSE - Test 10 (15 Nodes)

| Value 90/10 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 249.97 | 1936.48 | 13966.67 | 16153.10 | 5597.53 | 2.88 | 4.90 | 3.57 | 188.14 |
| Knapsack EK2 | 249.40 | 2062.34 | 24300.00 | 26611.74 | 7978.89 | 4.57 | 10.47 | 6.67 | 247.52 |
| Knapsack PFP1 | 259.57 | 2088.21 | 266.67 | 2614.45 | 993.57 | 2.95 | 5.27 | 0.07 | 206.68 |
| KnapsackPFP2 | 260.20 | 1972.90 | 7200.00 | 9433.10 | 2136.00 | 2.90 | 5.00 | 2.40 | 227.48 |
| Greedy EK1 | 249.83 | 1932.50 | 15300.00 | 17482.33 | 4228.88 | 2.88 | 4.90 | 3.90 | 164.26 |
| Greedy EK2 | 250.20 | 1985.60 | 26166.67 | 28402.47 | 6385.27 | 4.52 | 10.27 | 6.93 | 171.38 |
| Greedy PFP1 | 264.47 | 2014.61 | 2833.33 | 5112.41 | 3047.69 | 2.92 | 5.00 | 0.70 | 175.30 |
| Greedy PFP2 | 264.23 | 2006.64 | 8366.67 | 10637.55 | 5474.11 | 2.92 | 5.03 | 2.20 | 192.26 |

## Table 49. ACSE - Test 11 (15 Nodes)

| Value 95/05 Trans Rule: Combo | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 246.13 | 1914.29 | 16866.67 | 19027.09 | 355.17 | 2.88 | 4.57 | 4.13 | 211.99 |
| Knapsack EK2 | 248.40 | 1938.31 | 26000.00 | 28186.73 | 9084.98 | 4.45 | 10.00 | 6.87 | 232.97 |
| Knapsack PFP1 | 263.40 | 2001.09 | 1066.67 | 3331.16 | 1175.68 | 2.92 | 5.00 | 0.50 | 229.66 |
| Knapsack PFP2 | 260.43 | 2022.63 | 5866.67 | 8149.73 | 1122.99 | 2.92 | 5.43 | 1.47 | 263.75 |
| Greedy EK1 | 247.47 | 1925.00 | 14066.67 | 16239.13 | 5035.71 | 2.88 | 4.87 | 3.57 | 155.45 |
| Greedy EK2 | 247.80 | 1939.16 | 22500.00 | 24686.97 | 8070.09 | 4.44 | 10.30 | 6.10 | 162.79 |
| Greedy PFP1 | 261.23 | 1986.05 | 3633.33 | 5880.62 | 2652.01 | 2.91 | 4.97 | 1.00 | 179.42 |
| Greedy PFP2 | 261.40 | 2011.58 | 7766.67 | 10039.65 | 4886.40 | 2.92 | 5.00 | 2.00 | 196.87 |

## Table 50. ACSE - Test 12 (15 Nodes)

| Value 80/20 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 251.23 | 1889.72 | 14400.00 | 16540.95 | 4130.62 | 2.85 | 4.17 | 4.10 | 231.25 |
| Knapsack EK2 | 251.00 | 1971.00 | 17633.33 | 19855.33 | 9871.48 | 4.50 | 10.50 | 4.80 | 261.04 |
| Knapsack PFP1 | 254.47 | 1975.22 | 3833.33 | 6063.02 | 1465.43 | 2.91 | 4.80 | 1.03 | 260.03 |
| KnapsackPFP2 | 256.23 | 1999.69 | 10600.00 | 12855.92 | 5582.34 | 2.91 | 5.00 | 3.17 | 272.18 |
| Greedy EK1 | 249.77 | 1915.63 | 8400.00 | 10565.39 | 5866.76 | 2.88 | 5.00 | 2.10 | 163.24 |
| Greedy EK2 | 252.13 | 1964.13 | 14133.33 | 16349.61 | 6605.51 | 4.53 | 10.27 | 4.57 | 170.51 |
| Greedy PFP1 | 259.47 | 2022.23 | 800.00 | 3081.69 | 1619.31 | 2.91 | 5.00 | 0.20 | 176.11 |
| Greedy PFP2 | 254.97 | 2003.40 | 8500.00 | 10758.35 | 2019.52 | 2.91 | 5.00 | 2.90 | 217.13 |

## Table 51. ACSE - Test 13 (15 Nodes)

| Value 90/10 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 249.80 | 1911.01 | 7433.33 | 9594.15 | 5481.43 | 2.87 | 5.00 | 1.70 | 208.49 |
| Knapsack EK2 | 250.53 | 1973.95 | 16000.00 | 18224.48 | 8396.52 | 4.50 | 10.47 | 4.57 | 245.97 |
| Knapsack PFP1 | 252.67 | 1978.88 | 7266.67 | 9498.21 | 2717.56 | 2.91 | 5.00 | 1.87 | 248.89 |
| KnapsackPFP2 | 255.93 | 1999.40 | 5833.33 | 8088.66 | 2527.59 | 2.91 | 5.00 | 1.67 | 249.16 |
| Greedy EK1 | 248.70 | 1900.30 | 7266.67 | 9415.67 | 6018.47 | 2.86 | 4.27 | 1.70 | 167.86 |
| Greedy EK2 | 249.00 | 1926.14 | 16233.33 | 18408.48 | 8803.90 | 4.41 | 10.43 | 4.10 | 176.69 |
| Greedy PFP1 | 257.73 | 1992.08 | 1866.67 | 4116.48 | 2706.20 | 2.91 | 5.00 | 0.47 | 163.71 |
| Greedy PFP2 | 257.20 | 2010.31 | 9333.33 | 11600.86 | 2253.14 | 2.91 | 5.10 | 2.13 | 166.62 |

## Table 52. ACSE - Test 14 (15 Nodes)

| Value95/05 Trans Rule: Fixed | Fixed Edge Cost | Comm Flow Cost | Penalty | Total Cost | SD | Num of Hops | Net Diam | Drop Comm | Run Time |
|---|---|---|---|---|---|---|---|---|---|
| Knapsack EK1 | 249.57 | 1900.39 | 11666.67 | 13816.62 | 5765.39 | 2.87 | 4.83 | 3.53 | 224.87 |
| Knapsack EK2 | 249.00 | 1907.00 | 23300.00 | 25455.99 | 8297.55 | 4.41 | 10.47 | 6.03 | 285.32 |
| Knapsack PFP1 | 262.90 | 1989.66 | 3600.00 | 5852.56 | 2623.96 | 2.89 | 5.00 | 0.90 | 236.34 |
| Knapsack PFP2 | 255.07 | 2013.85 | 3900.00 | 6168.92 | 4127.89 | 2.93 | 5.00 | 1.13 | 269.13 |
| Greedy EK1 | 249.17 | 1920.14 | 7466.67 | 9635.96 | 5839.35 | 2.86 | 5.00 | 1.80 | 158.13 |
| Greedy EK2 | 248.83 | 1956.70 | 15233.33 | 17438.87 | 7802.50 | 4.43 | 10.63 | 4.07 | 161.73 |
| Greedy PFP1 | 250.70 | 1993.89 | 3766.67 | 6011.26 | 3107.38 | 2.92 | 5.07 | 1.03 | 162.81 |
| Greedy PFP2 | 254.43 | 1998.90 | 8733.33 | 10986.67 | 5160.41 | 2.91 | 5.00 | 2.13 | 165.13 |

## Appendix C: Phase 2 (Dynamic) Results - 10 Node Network

### Table 53. Dynamic ACSE Solution Cost Results - File 1 (10 Nodes)

| Solution Cost Results - 10 Node ACSE (File 1) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 |
| **ACSE-1** | 6184.77 | 7.02 | 6184.38 | 6.62 | 6482.06 | 1635.05 | 6481.61 | 1631.92 |
| **ACSE-2** | 6181.91 | 6.62 | 6181.68 | 5.85 | 6180.45 | 7.07 | 6781.94 | 2269.40 |
| **ACSE-3** | 6481.84 | 1633.58 | 6316.90 | 1893.39 | 6779.09 | 2271.48 | 5815.23 | 2451.08 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 471.25 | 0.00 | 5414.67 | 0.00 | 402.00 | 0.00 |
| **ACSE-1** | 6182.93 | 5.16 | 1210.87 | 28.42 | 1195.15 | 8.26 | 2438.53 | 2245.64 |
| **ACSE-2** | 6182.47 | 4.98 | 1207.03 | 27.00 | 3808.13 | 4964.88 | 7189.04 | 8373.28 |
| **ACSE-3** | 6320.07 | 1896.48 | 1533.94 | 1799.77 | 1691.91 | 2720.70 | 1776.56 | 1734.18 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 612.00 | 0.00 | 52380.00 | 0.00 | 56411.00 | 0.00 |
| **ACSE-1** | 6548.67 | 1639.28 | 25023.32 | 1067.13 | 44974.55 | 725.01 | 24990.51 | 8.14 |
| **ACSE-2** | 6252.80 | 6.25 | 24746.52 | 1025.49 | 44055.12 | 1790.19 | 25687.16 | 2541.58 |
| **ACSE-3** | 6683.77 | 2360.60 | 25219.78 | 1455.55 | 43915.98 | 1850.59 | 24991.89 | 10.04 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 56331.70 | 0.00 | 31446.80 | 0.00 | 49381.00 | 0.00 |
| **ACSE-1** | 6279.45 | 6.66 | 44020.55 | 3866.77 | 47965.90 | 41.21 | 43836.31 | 14372.03 |
| **ACSE-2** | 6281.80 | 7.72 | 41277.11 | 4057.29 | 47976.86 | 43.28 | 28791.30 | 11935.45 |
| **ACSE-3** | 6112.80 | 905.36 | 42367.07 | 2296.76 | 49180.13 | 3421.14 | 44724.53 | 14960.88 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 487.00 | 0.00 | 62357.00 | 0.00 | 74354.20 | 0.00 |
| **ACSE-1** | 6203.72 | 9.86 | 8393.54 | 5254.77 | 45117.18 | 3827.73 | 76796.47 | 6.78 |
| **ACSE-2** | 6202.92 | 10.14 | 10284.68 | 6280.54 | 44703.63 | 4414.03 | 76773.67 | 3.79 |
| **ACSE-3** | 6174.60 | 2110.07 | 8414.18 | 6446.14 | 46481.71 | 3963.08 | 76795.33 | 7.66 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 14439.00 | 0.00 | 20375.00 | 0.00 | 27360.70 | 0.00 |
| **ACSE-1** | 6193.14 | 10.61 | 1341.93 | 38.83 | 31133.99 | 908.60 | 40853.50 | 14455.93 |
| **ACSE-2** | 6194.90 | 10.26 | 1293.40 | 21.28 | 30967.70 | 1.52 | 36739.94 | 15647.01 |
| **ACSE-3** | 6461.66 | 2561.65 | 1317.17 | 35.29 | 31299.87 | 1262.60 | 37966.95 | 15644.46 |

**Table 54. Dynamic ACSE Solution Cost Results - File 2 (10 Nodes)**

| Solution Cost Results - 10 Node ACSE (File 2) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 |
| **ACSE-1** | 1987.23 | 1782.73 | 3597.34 | 1853.27 | 3272.09 | 1736.31 | 3273.12 | 1732.29 |
| **ACSE-2** | 3174.25 | 1996.57 | 2910.97 | 2223.45 | 3143.50 | 2388.00 | 2449.22 | 1713.88 |
| **ACSE-3** | 2473.73 | 2695.20 | 4039.82 | 3642.34 | 2529.72 | 1980.57 | 3546.56 | 1983.11 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 471.25 | 0.00 | 5414.67 | 0.00 | 402.00 | 0.00 |
| **ACSE-1** | 3537.09 | 2392.90 | 6259.81 | 7890.10 | 8225.08 | 3760.56 | 11369.79 | 4718.72 |
| **ACSE-2** | 2711.55 | 2017.29 | 5912.41 | 6693.79 | 7764.57 | 3315.59 | 11183.11 | 4107.36 |
| **ACSE-3** | 2200.76 | 1789.43 | 6700.44 | 7005.05 | 8833.53 | 4440.10 | 9900.19 | 4268.35 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 612.00 | 0.00 | 52380.00 | 0.00 | 56411.00 | 0.00 |
| **ACSE-1** | 2807.08 | 1856.76 | 1371.74 | 784.46 | 9239.15 | 4175.04 | 41734.78 | 4144.35 |
| **ACSE-2** | 2660.73 | 1816.44 | 2494.43 | 1869.45 | 8154.87 | 4626.53 | 45078.13 | 5043.59 |
| **ACSE-3** | 2832.93 | 2101.17 | 1870.10 | 1440.87 | 8255.29 | 3844.56 | 41769.54 | 4450.17 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 56331.70 | 0.00 | 31446.80 | 0.00 | 49381.00 | 0.00 |
| **ACSE-1** | 3035.29 | 1768.11 | 10621.01 | 4445.08 | 6961.03 | 7196.26 | 43913.05 | 1510.64 |
| **ACSE-2** | 2699.83 | 2213.27 | 10024.53 | 4548.98 | 5565.68 | 7030.06 | 44620.59 | 1149.25 |
| **ACSE-3** | 2755.41 | 2665.32 | 9597.11 | 4726.52 | 6704.55 | 5803.89 | 43541.74 | 1581.59 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 487.00 | 0.00 | 62357.00 | 0.00 | 74354.20 | 0.00 |
| **ACSE-1** | 2675.45 | 1828.81 | 21019.99 | 3123.46 | 26164.50 | 0.00 | 73806.00 | 29.41 |
| **ACSE-2** | 2842.66 | 1777.96 | 21723.50 | 1997.03 | 26164.50 | 0.00 | 73814.43 | 25.25 |
| **ACSE-3** | 2558.88 | 1981.96 | 21903.32 | 3660.25 | 26164.50 | 0.00 | 73810.00 | 30.24 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 14439.00 | 0.00 | 20375.00 | 0.00 | 27360.70 | 0.00 |
| **ACSE-1** | 2437.22 | 1976.04 | 9145.18 | 4964.05 | 26087.35 | 1844.76 | 51021.50 | 0.00 |
| **ACSE-2** | 2641.81 | 1973.72 | 7386.25 | 4364.08 | 33166.99 | 8284.16 | 51021.50 | 0.00 |
| **ACSE-3** | 2370.81 | 1784.48 | 10307.15 | 4978.54 | 26846.27 | 1727.58 | 51021.50 | 0.00 |

**Table 55. Dynamic ACSE Solution Cost Results - File 3 (10 Nodes)**

| Solution Cost Results - 10 Node ACSE (File 3) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 372.40 | 0.00 | 372.40 | 0.00 | 372.40 | 0.00 |
| **ACSE-1** | 3548.15 | 5698.83 | 1037.93 | 17.08 | 1034.13 | 22.26 | 1036.73 | 10.46 |
| **ACSE-2** | 2042.88 | 3810.77 | 5717.49 | 7316.62 | 2037.07 | 3819.25 | 1537.13 | 2748.06 |
| **ACSE-3** | 2038.80 | 3818.75 | 1020.07 | 34.26 | 1191.03 | 922.80 | 1651.24 | 2811.96 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 339.60 | 0.00 | 350.67 | 0.00 | 345.50 | 0.00 |
| **ACSE-1** | 2720.00 | 5156.42 | 1362.52 | 1450.17 | 1259.26 | 901.88 | 1082.23 | 3.38 |
| **ACSE-2** | 1547.80 | 2747.63 | 1558.39 | 1676.07 | 1104.24 | 32.46 | 1083.90 | 7.44 |
| **ACSE-3** | 3217.34 | 5692.61 | 1093.73 | 56.84 | 1076.94 | 43.71 | 1075.30 | 19.20 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 383.60 | 0.00 | 30398.00 | 0.00 | 28456.80 | 0.00 |
| **ACSE-1** | 2617.77 | 4593.16 | 1046.80 | 18.32 | 33602.03 | 2522.24 | 28891.70 | 7.06 |
| **ACSE-2** | 2288.51 | 3878.99 | 1047.29 | 14.52 | 34601.10 | 2240.12 | 28889.77 | 6.07 |
| **ACSE-3** | 5290.35 | 7095.37 | 1035.03 | 19.75 | 34426.47 | 2313.94 | 28895.90 | 11.28 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 368.10 | 0.00 | 399.00 | 0.00 | 27353.80 | 0.00 |
| **ACSE-1** | 3430.99 | 5209.27 | 1036.23 | 14.81 | 3108.39 | 3055.35 | 28086.23 | 1.46 |
| **ACSE-2** | 2094.32 | 3811.93 | 1046.37 | 26.08 | 3433.13 | 2281.86 | 28086.50 | 0.00 |
| **ACSE-3** | 3428.75 | 5371.69 | 1032.17 | 16.06 | 2829.23 | 2363.72 | 28087.03 | 2.92 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 447.40 | 0.00 | 37420.80 | 0.00 | 107326.00 | 0.00 |
| **ACSE-1** | 2984.27 | 4827.30 | 5546.16 | 5130.46 | 8787.46 | 4846.77 | 82783.00 | 0.00 |
| **ACSE-2** | 2310.69 | 3866.62 | 6417.31 | 5975.09 | 5053.93 | 4328.83 | 82783.00 | 0.00 |
| **ACSE-3** | 4308.91 | 5504.79 | 5106.24 | 4965.58 | 6956.33 | 5350.40 | 82783.00 | 0.00 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 395.00 | 0.00 | 431.00 | 0.00 | 81322.00 | 0.00 |
| **ACSE-1** | 2143.11 | 3802.73 | 1817.99 | 4372.38 | 26138.17 | 9.37 | 116324.58 | 12953.34 |
| **ACSE-2** | 2142.57 | 3802.83 | 6413.65 | 8645.59 | 26120.90 | 4.85 | 123734.98 | 5077.52 |
| **ACSE-3** | 2798.65 | 4604.94 | 3222.33 | 6712.63 | 26132.20 | 12.64 | 111693.08 | 14101.38 |

**Table 56. Dynamic ACSE Solution Cost Results - File 4 (10 Nodes)**

| Solution Cost Results - 10 Node ACSE (File 4) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 21410.00 | 0.00 | 21410.00 | 0.00 | 21410.00 | 0.00 |
| **ACSE-1** | 9932.99 | 6519.35 | 9723.07 | 6547.15 | 8551.36 | 5961.74 | 11707.02 | 6949.05 |
| **ACSE-2** | 12594.07 | 6101.79 | 10473.09 | 5658.07 | 10074.99 | 5533.67 | 10551.69 | 5987.73 |
| **ACSE-3** | 9870.98 | 5429.32 | 10846.00 | 7000.02 | 10244.98 | 7743.78 | 9302.31 | 5950.31 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 517.00 | 0.00 | 416.00 | 0.00 | 459.00 | 0.00 |
| **ACSE-1** | 9999.32 | 7507.26 | 11849.99 | 7152.19 | 7953.22 | 5904.23 | 1089.17 | 13.91 |
| **ACSE-2** | 10397.17 | 5644.81 | 13217.51 | 5830.44 | 5022.46 | 5426.17 | 1100.68 | 19.63 |
| **ACSE-3** | 9164.50 | 6514.68 | 10708.75 | 8731.19 | 4319.98 | 4776.32 | 1093.62 | 29.36 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 22416.20 | 0.00 | 25462.00 | 0.00 | 23442.40 | 0.00 |
| **ACSE-1** | 9569.96 | 4206.46 | 18262.36 | 2436.32 | 1176.55 | 0.00 | 28408.98 | 4697.27 |
| **ACSE-2** | 9467.19 | 7534.82 | 17351.17 | 4677.64 | 1173.18 | 4.84 | 29152.67 | 4184.96 |
| **ACSE-3** | 9637.16 | 4642.59 | 18762.47 | 2235.15 | 1176.21 | 1.84 | 26877.83 | 3551.20 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 554.60 | 0.00 | 36434.40 | 0.00 | 91353.20 | 0.00 |
| **ACSE-1** | 10908.51 | 6509.77 | 12986.00 | 2591.74 | 19277.72 | 6837.07 | 81202.43 | 3649.04 |
| **ACSE-2** | 11685.87 | 6382.00 | 11794.32 | 3906.88 | 18846.26 | 9613.80 | 82828.81 | 4023.94 |
| **ACSE-3** | 8979.97 | 5473.24 | 14277.84 | 2266.76 | 21061.93 | 6973.81 | 82672.87 | 3824.63 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 14439.50 | 0.00 | 51358.00 | 0.00 | 38390.00 | 0.00 |
| **ACSE-1** | 10092.72 | 4069.63 | 31457.33 | 6154.47 | 37045.44 | 2710.78 | 62095.25 | 177.51 |
| **ACSE-2** | 10852.15 | 6149.07 | 31378.15 | 6260.99 | 37070.63 | 561.58 | 62127.67 | 0.51 |
| **ACSE-3** | 9341.74 | 5219.89 | 34970.36 | 7270.02 | 37430.10 | 2446.88 | 62094.99 | 177.46 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 26512.40 | 0.00 | 53361.00 | 0.00 | 76328.40 | 0.00 |
| **ACSE-1** | 11801.63 | 7015.83 | 1228.72 | 15.37 | 70530.92 | 3397.57 | 88407.14 | 2508.42 |
| **ACSE-2** | 11194.62 | 5068.85 | 1228.85 | 16.40 | 78404.69 | 9974.70 | 87402.90 | 1237.31 |
| **ACSE-3** | 9732.25 | 5731.56 | 1229.95 | 22.14 | 76315.84 | 8185.89 | 87914.08 | 2457.58 |

**Table 57. Dynamic ACSE Solution Cost Results - File 5 (10 Nodes)**

| Solution Cost Results - 10 Node ACSE (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 466.47 | 0.00 | 466.47 | 0.00 | 466.47 | 0.00 |
| ACSE-1 | 1078.60 | 26.95 | 1061.17 | 7.62 | 1061.00 | 7.61 | 1059.60 | 2.28 |
| ACSE-2 | 1070.53 | 22.03 | 1074.07 | 26.22 | 1080.53 | 26.89 | 1085.27 | 29.00 |
| ACSE-3 | 1081.83 | 29.44 | 1062.18 | 11.49 | 1059.57 | 7.77 | 1059.67 | 9.59 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 468.25 | 0.00 | 464.30 | 0.00 | 476.30 | 0.00 |
| ACSE-1 | 1141.00 | 29.11 | 10593.46 | 1820.18 | 1189.27 | 12.78 | 1219.80 | 30.73 |
| ACSE-2 | 1144.73 | 29.95 | 12526.68 | 3028.80 | 1221.00 | 56.55 | 1193.70 | 62.89 |
| ACSE-3 | 1138.40 | 27.02 | 10854.22 | 1836.35 | 1186.77 | 8.46 | 1233.37 | 29.99 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 520.25 | 0.00 | 55393.80 | 0.00 | 71366.00 | 0.00 |
| ACSE-1 | 1117.37 | 13.53 | 6172.20 | 7.46 | 20222.64 | 9938.85 | 31859.18 | 10681.10 |
| ACSE-2 | 1113.70 | 11.64 | 3842.07 | 2538.83 | 23077.42 | 6527.83 | 35169.74 | 12169.60 |
| ACSE-3 | 1118.20 | 14.16 | 6174.93 | 5.37 | 21552.41 | 10121.23 | 30203.90 | 9411.48 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 474.60 | 0.00 | 15418.70 | 0.00 | 535.00 | 0.00 |
| ACSE-1 | 1110.83 | 20.97 | 9329.79 | 1874.15 | 18190.00 | 0.00 | 9396.20 | 4.57 |
| ACSE-2 | 1113.67 | 21.10 | 12677.88 | 2184.21 | 18190.00 | 0.00 | 9395.60 | 3.29 |
| ACSE-3 | 1107.90 | 20.30 | 9401.38 | 1783.64 | 18190.00 | 0.00 | 9395.60 | 3.29 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 33424.00 | 0.00 | 32391.00 | 0.00 | 58353.00 | 0.00 |
| ACSE-1 | 1090.47 | 35.10 | 1224.67 | 10.96 | 33066.20 | 14.45 | 86683.00 | 0.00 |
| ACSE-2 | 1099.07 | 39.93 | 1228.87 | 14.83 | 33858.65 | 1339.65 | 86683.00 | 0.00 |
| ACSE-3 | 1106.97 | 38.71 | 1222.87 | 11.34 | 33161.06 | 544.46 | 86683.00 | 0.00 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 28388.00 | 0.00 | 10434.00 | 0.00 | 77318.70 | 0.00 |
| ACSE-1 | 1081.80 | 34.69 | 29138.75 | 4.02 | 59822.23 | 4.75 | 104614.00 | 0.00 |
| ACSE-2 | 1072.30 | 31.26 | 29148.71 | 17.76 | 59831.40 | 12.10 | 104614.00 | 0.00 |
| ACSE-3 | 1078.27 | 33.71 | 29139.72 | 4.42 | 59821.90 | 3.29 | 104614.00 | 0.00 |

**Table 58. Dynamic ACSE Solution Cost Results - File 6 (10 Nodes)**

| Solution Cost Results - 10 Node ACSE (File 6) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 367.00 | 0.00 | 367.00 | 0.00 | 367.00 | 0.00 | 367.00 | 0.00 |
| ACSE-1 | 4037.34 | 3290.41 | 4466.10 | 3497.33 | 3919.11 | 3190.60 | 3852.81 | 3389.63 |
| ACSE-2 | 3544.11 | 2974.23 | 4408.37 | 3477.32 | 4150.41 | 3228.30 | 4132.14 | 3172.70 |
| ACSE-3 | 4069.00 | 3440.41 | 3949.39 | 3840.70 | 3248.15 | 3179.67 | 2916.35 | 2671.52 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 367.00 | 0.00 | 352.00 | 0.00 | 368.00 | 0.00 | 356.50 | 0.00 |
| ACSE-1 | 4900.76 | 3836.31 | 1829.72 | 1899.67 | 3187.03 | 2514.64 | 12367.38 | 7359.92 |
| ACSE-2 | 3251.16 | 2984.06 | 1689.18 | 1723.16 | 2387.36 | 2240.19 | 11902.93 | 8029.99 |
| ACSE-3 | 5218.77 | 4397.35 | 1173.69 | 912.17 | 3649.95 | 3541.68 | 11265.93 | 8836.08 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 367.00 | 0.00 | 391.50 | 0.00 | 427.75 | 0.00 | 21424.00 | 0.00 |
| ACSE-1 | 4957.91 | 4244.67 | 1155.47 | 3.10 | 1034.37 | 8.62 | 47934.38 | 17.14 |
| ACSE-2 | 5157.53 | 4317.21 | 3706.90 | 4397.85 | 1052.23 | 18.05 | 50105.09 | 2529.73 |
| ACSE-3 | 4751.98 | 3738.89 | 1488.70 | 1832.87 | 1032.07 | 13.56 | 47931.28 | 26.44 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 367.00 | 0.00 | 369.50 | 0.00 | 443.40 | 0.00 | 6483.60 | 0.00 |
| ACSE-1 | 4109.01 | 3380.72 | 8315.31 | 3837.20 | 3627.43 | 2603.85 | 8204.63 | 8153.59 |
| ACSE-2 | 3147.18 | 3136.15 | 10138.21 | 1329.42 | 5848.67 | 2050.02 | 7722.38 | 8139.68 |
| ACSE-3 | 3675.95 | 3519.93 | 7237.43 | 4294.66 | 3110.75 | 2517.44 | 7200.14 | 8854.50 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 367.00 | 0.00 | 402.60 | 0.00 | 442.60 | 0.00 | 121214.00 | 0.00 |
| ACSE-1 | 4487.54 | 4150.22 | 1101.18 | 41.01 | 14802.17 | 2350.47 | 135565.80 | 7.15 |
| ACSE-2 | 3980.46 | 3866.63 | 1248.97 | 726.01 | 16893.73 | 1180.36 | 135596.87 | 22.98 |
| ACSE-3 | 4459.62 | 4541.31 | 1089.12 | 34.18 | 14812.17 | 3564.22 | 135565.33 | 4.11 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 367.00 | 0.00 | 396.20 | 0.00 | 21415.00 | 0.00 | 45384.00 | 0.00 |
| ACSE-1 | 5157.04 | 3482.80 | 1387.58 | 14.51 | 22273.27 | 9948.24 | 51052.00 | 0.00 |
| ACSE-2 | 4723.60 | 4016.24 | 1394.88 | 20.25 | 22098.89 | 6779.65 | 51104.80 | 32.38 |
| ACSE-3 | 4529.65 | 3704.34 | 1366.15 | 30.25 | 23921.91 | 10021.17 | 51052.00 | 0.00 |

**Table 59. Dynamic ACSE Solution Cost Results - File 7 (10 Nodes)**

| Solution Cost Results - 10 Node ACSE (File 7) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 369.00 | 0.00 | 369.00 | 0.00 | 369.00 | 0.00 | 369.00 | 0.00 |
| ACSE-1 | 5389.52 | 4704.60 | 5697.97 | 5834.45 | 7381.99 | 6770.93 | 5528.64 | 6203.16 |
| ACSE-2 | 5751.72 | 4557.24 | 6242.72 | 5691.03 | 7333.13 | 5790.34 | 8660.40 | 4803.69 |
| ACSE-3 | 4452.78 | 4566.14 | 6421.41 | 6024.69 | 4636.46 | 4652.38 | 7379.24 | 6342.11 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 369.00 | 0.00 | 350.00 | 0.00 | 393.50 | 0.00 | 371.00 | 0.00 |
| ACSE-1 | 6922.39 | 4638.42 | 2059.50 | 1802.78 | 5640.31 | 1254.01 | 10270.53 | 2528.69 |
| ACSE-2 | 5924.18 | 5026.53 | 1800.23 | 1630.36 | 5603.18 | 1055.71 | 9946.00 | 3046.49 |
| ACSE-3 | 6876.98 | 5483.43 | 1923.30 | 1964.36 | 4807.78 | 2137.29 | 8321.63 | 5061.30 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 369.00 | 0.00 | 27319.00 | 0.00 | 25377.00 | 0.00 | 20415.40 | 0.00 |
| ACSE-1 | 6943.13 | 4546.59 | 27842.07 | 17.31 | 25996.47 | 2.08 | 20935.10 | 13.09 |
| ACSE-2 | 5115.54 | 5603.41 | 27869.07 | 62.14 | 26011.95 | 25.55 | 20941.14 | 14.26 |
| ACSE-3 | 5418.90 | 4621.67 | 27829.83 | 26.57 | 26003.81 | 23.04 | 20938.14 | 20.31 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 369.00 | 0.00 | 366.50 | 0.00 | 452.00 | 0.00 | 410.00 | 0.00 |
| ACSE-1 | 6298.80 | 4878.18 | 1077.59 | 17.47 | 4001.70 | 2102.25 | 6234.15 | 0.00 |
| ACSE-2 | 5955.61 | 5408.89 | 1072.58 | 17.56 | 6964.28 | 5551.74 | 6234.15 | 0.00 |
| ACSE-3 | 5827.55 | 4685.87 | 1083.54 | 21.42 | 4619.92 | 2491.17 | 6234.15 | 0.00 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 369.00 | 0.00 | 395.40 | 0.00 | 492.00 | 0.00 | 486.00 | 0.00 |
| ACSE-1 | 5971.32 | 4556.28 | 1060.97 | 22.53 | 13954.72 | 5580.58 | 40087.51 | 5951.26 |
| ACSE-2 | 6059.16 | 4901.00 | 1050.47 | 30.96 | 10900.29 | 5044.63 | 39995.00 | 10384.92 |
| ACSE-3 | 2934.22 | 3622.73 | 1048.31 | 21.55 | 10653.74 | 4731.69 | 41815.02 | 4471.02 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 369.00 | 0.00 | 24362.80 | 0.00 | 26458.00 | 0.00 | 104270.00 | 0.00 |
| ACSE-1 | 5541.21 | 5766.60 | 26457.39 | 5104.26 | 20024.72 | 9.15 | 156467.00 | 0.00 |
| ACSE-2 | 5775.98 | 4716.14 | 20686.67 | 9620.56 | 20026.35 | 3.37 | 156467.00 | 0.00 |
| ACSE-3 | 8058.25 | 5729.43 | 25355.92 | 6859.81 | 20028.95 | 17.29 | 156467.00 | 0.00 |

Table 60. Dynamic ACSE Solution Cost Results - File 8 (10 Nodes)

| Solution Cost Results - 10 Node ACSE (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 11409.00 | 0.00 | 11409.00 | 0.00 | 11409.00 | 0.00 |
| **ACSE-1** | 2816.46 | 2655.93 | 1747.81 | 1606.60 | 1883.03 | 1702.57 | 2410.13 | 1939.92 |
| **ACSE-2** | 2815.38 | 2359.19 | 1996.57 | 2147.88 | 1578.83 | 1363.70 | 1592.21 | 1912.25 |
| **ACSE-3** | 1978.86 | 2127.17 | 1751.82 | 1618.33 | 2153.31 | 1843.43 | 2013.91 | 1784.89 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 11371.00 | 0.00 | 37331.40 | 0.00 | 35334.00 | 0.00 |
| **ACSE-1** | 2144.67 | 1756.84 | 5777.06 | 8340.68 | 37937.30 | 30.07 | 35848.71 | 12.69 |
| **ACSE-2** | 2011.58 | 1686.09 | 7855.11 | 8376.00 | 37957.20 | 38.87 | 35866.07 | 46.60 |
| **ACSE-3** | 2604.48 | 2343.81 | 6139.92 | 9390.73 | 37934.03 | 28.06 | 35844.03 | 17.45 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 426.00 | 0.00 | 4427.00 | 0.00 | 36367.00 | 0.00 |
| **ACSE-1** | 2408.71 | 2380.16 | 972.03 | 0.51 | 12103.37 | 14.11 | 28543.17 | 2724.04 |
| **ACSE-2** | 1990.73 | 2119.97 | 1571.69 | 1820.87 | 12104.60 | 15.49 | 28206.90 | 906.37 |
| **ACSE-3** | 2622.66 | 2354.38 | 1165.99 | 1088.97 | 12100.57 | 14.21 | 28867.50 | 2945.15 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 420.80 | 0.00 | 13401.00 | 0.00 | 5445.00 | 0.00 |
| **ACSE-1** | 2565.88 | 2372.36 | 1449.84 | 2189.56 | 28168.07 | 0.37 | 64976.07 | 7.48 |
| **ACSE-2** | 2254.67 | 2231.62 | 2917.61 | 3274.64 | 32394.10 | 8990.30 | 62718.23 | 2007.50 |
| **ACSE-3** | 2279.09 | 2243.94 | 1849.74 | 3049.17 | 26786.37 | 4130.63 | 64577.20 | 1217.09 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 8418.00 | 0.00 | 454.00 | 0.00 | 30441.00 | 0.00 |
| **ACSE-1** | 2194.12 | 1853.51 | 1113.93 | 60.26 | 1460.54 | 1275.21 | 3320.20 | 1435.51 |
| **ACSE-2** | 2152.55 | 2560.20 | 1291.07 | 1119.23 | 3172.66 | 2520.62 | 6405.47 | 14120.33 |
| **ACSE-3** | 2155.81 | 2212.32 | 1052.94 | 27.71 | 1125.59 | 20.61 | 2623.23 | 1507.56 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 463.00 | 0.00 | 76254.00 | 0.00 | 108289.00 | 0.00 |
| **ACSE-1** | 1715.28 | 1494.36 | 14727.21 | 7657.07 | 39050.72 | 2486.03 | 103982.60 | 1902.59 |
| **ACSE-2** | 2384.50 | 1897.74 | 15023.68 | 5159.75 | 37887.18 | 1891.08 | 103030.13 | 1035.64 |
| **ACSE-3** | 1734.85 | 1527.87 | 9307.93 | 5752.32 | 39549.82 | 2537.19 | 103302.27 | 1411.33 |

**Table 61. Dynamic ACSE Solution Cost Results - File 9 (10 Nodes)**

| Solution Cost Results - 10 Node ACSE (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 410.00 | 0.00 | 410.00 | 0.00 | 410.00 | 0.00 |
| **ACSE-1** | 1071.37 | 20.98 | 1064.37 | 26.26 | 1066.47 | 33.82 | 1065.90 | 32.95 |
| **ACSE-2** | 1066.77 | 27.90 | 1059.13 | 17.32 | 1068.53 | 26.56 | 1064.60 | 20.38 |
| **ACSE-3** | 1061.17 | 34.61 | 1045.23 | 42.07 | 1048.08 | 41.65 | 1382.74 | 1267.69 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 384.40 | 0.00 | 355.83 | 0.00 | 321.00 | 0.00 |
| **ACSE-1** | 1018.57 | 27.50 | 987.47 | 11.83 | 1051.13 | 28.68 | 1023.17 | 19.47 |
| **ACSE-2** | 1019.23 | 24.04 | 986.97 | 9.39 | 1069.33 | 22.78 | 1034.27 | 22.84 |
| **ACSE-3** | 1149.60 | 735.80 | 977.97 | 20.05 | 1612.14 | 3091.08 | 1402.15 | 1599.18 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 430.00 | 0.00 | 24529.80 | 0.00 | 22454.80 | 0.00 |
| **ACSE-1** | 1105.00 | 14.29 | 9817.39 | 8106.26 | 27770.97 | 3634.16 | 26114.71 | 1.60 |
| **ACSE-2** | 1104.43 | 13.98 | 11424.82 | 8424.99 | 27479.38 | 2006.56 | 26112.37 | 4.83 |
| **ACSE-3** | 1264.26 | 916.52 | 10951.16 | 9053.59 | 27372.03 | 1458.05 | 26208.65 | 543.71 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 392.75 | 0.00 | 432.00 | 0.00 | 531.80 | 0.00 |
| **ACSE-1** | 1117.30 | 15.72 | 1131.08 | 16.95 | 2854.04 | 3022.67 | 6628.09 | 2902.46 |
| **ACSE-2** | 1118.87 | 13.96 | 1132.82 | 26.35 | 3676.48 | 3924.44 | 13668.20 | 4653.27 |
| **ACSE-3** | 1122.82 | 30.73 | 1796.66 | 1735.85 | 3018.38 | 3064.16 | 6228.44 | 4309.84 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 426.00 | 0.00 | 547.25 | 0.00 | 89378.00 | 0.00 |
| **ACSE-1** | 1091.67 | 34.21 | 4286.23 | 4377.86 | 1268.33 | 60.12 | 58062.00 | 9.15 |
| **ACSE-2** | 3073.43 | 7573.42 | 8035.67 | 8031.17 | 1299.92 | 80.34 | 57090.60 | 3762.63 |
| **ACSE-3** | 1094.07 | 33.25 | 4154.99 | 4219.64 | 1251.44 | 57.86 | 57783.37 | 1028.68 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 437.00 | 0.00 | 21401.00 | 0.00 | 92267.00 | 0.00 |
| **ACSE-1** | 1063.13 | 23.08 | 11399.66 | 5134.25 | 27088.07 | 11.47 | 125711.00 | 0.00 |
| **ACSE-2** | 1072.93 | 22.35 | 10361.75 | 3744.17 | 27127.23 | 11.06 | 125711.00 | 0.00 |
| **ACSE-3** | 1197.23 | 726.11 | 10482.66 | 5572.87 | 27097.80 | 15.92 | 125711.00 | 0.00 |

**Table 62. Dynamic ACSE Solution Cost Results - File 10 (10 Nodes)**

| Solution Cost Results - 10 Node ACSE (File 10) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 30328.40 | 0.00 | 30328.40 | 0.00 | 30328.40 | 0.00 |
| **ACSE-1** | 30868.53 | 33.48 | 30865.90 | 38.55 | 30870.23 | 34.73 | 30869.37 | 34.71 |
| **ACSE-2** | 30868.09 | 33.97 | 30861.20 | 36.68 | 30866.37 | 35.18 | 30868.77 | 33.74 |
| **ACSE-3** | 30868.50 | 37.00 | 30852.90 | 37.25 | 30871.97 | 35.30 | 30855.40 | 36.44 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 29334.00 | 0.00 | 26342.60 | 0.00 | 26360.30 | 0.00 |
| **ACSE-1** | 30905.40 | 33.13 | 32034.13 | 2514.31 | 29145.95 | 2278.30 | 40719.17 | 1984.44 |
| **ACSE-2** | 30898.41 | 42.49 | 32051.45 | 2502.65 | 28867.80 | 2239.61 | 40394.26 | 1875.83 |
| **ACSE-3** | 30915.48 | 38.84 | 32023.91 | 2513.32 | 29079.37 | 2220.82 | 41244.48 | 1985.29 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 377.00 | 0.00 | 479.40 | 0.00 | 52338.00 | 0.00 |
| **ACSE-1** | 30857.73 | 42.53 | 916.71 | 11.67 | 6667.40 | 8121.02 | 22057.17 | 5.27 |
| **ACSE-2** | 30835.85 | 29.62 | 926.77 | 21.31 | 5603.39 | 7932.50 | 22064.97 | 11.07 |
| **ACSE-3** | 30854.11 | 50.02 | 912.53 | 14.00 | 4410.87 | 6891.08 | 22586.90 | 2900.50 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 382.40 | 0.00 | 388.00 | 0.00 | 30351.20 | 0.00 |
| **ACSE-1** | 30943.40 | 36.56 | 23476.70 | 3285.71 | 8503.84 | 5095.10 | 1123.70 | 13.31 |
| **ACSE-2** | 30960.30 | 39.16 | 24075.17 | 2.15 | 11879.79 | 5932.33 | 1136.87 | 23.31 |
| **ACSE-3** | 30950.17 | 45.59 | 19260.60 | 8088.41 | 8741.17 | 6226.20 | 1125.27 | 18.85 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 35389.00 | 0.00 | 450.00 | 0.00 | 22495.00 | 0.00 |
| **ACSE-1** | 30914.13 | 36.10 | 35984.30 | 24.42 | 1164.00 | 8.63 | 43254.07 | 18.63 |
| **ACSE-2** | 28862.23 | 7845.70 | 33574.83 | 9126.68 | 1085.60 | 295.21 | 40377.13 | 10975.73 |
| **ACSE-3** | 30909.87 | 40.97 | 35983.09 | 23.50 | 1161.73 | 7.27 | 43254.07 | 18.63 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 419.67 | 0.00 | 29429.00 | 0.00 | 63365.60 | 0.00 |
| **ACSE-1** | 30977.93 | 42.09 | 2830.30 | 1556.76 | 30279.06 | 44.11 | 106828.00 | 0.00 |
| **ACSE-2** | 30965.60 | 34.65 | 2915.24 | 3104.51 | 30516.11 | 912.59 | 106828.00 | 0.00 |
| **ACSE-3** | 30971.80 | 35.33 | 3000.05 | 1621.29 | 32043.99 | 7837.13 | 106828.00 | 0.00 |

## Table 63. Dynamic ACSS Solution Cost Results - File 1 (10 Nodes)

| Solution Cost Results - 10 Node ACSS (File 1) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 |
| **ACSS-1** | 1096.67 | 27.33 | 1099.69 | 38.78 | 1083.55 | 37.62 | 1092.98 | 44.59 |
| **ACSS-2** | 1093.33 | 34.95 | 1100.54 | 40.38 | 1090.92 | 35.09 | 1094.96 | 20.08 |
| **ACSS-3** | 1051.00 | 40.34 | 1049.42 | 43.40 | 1046.03 | 38.68 | 1029.28 | 33.22 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 471.25 | 0.00 | 5414.67 | 0.00 | 402.00 | 0.00 |
| **ACSS-1** | 1100.22 | 32.91 | 1100.22 | 32.91 | 1034.33 | 18.87 | 1078.01 | 28.90 |
| **ACSS-2** | 1108.27 | 30.39 | 1034.18 | 18.57 | 1073.40 | 31.93 | 982.23 | 19.90 |
| **ACSS-3** | 1058.65 | 40.36 | 986.97 | 50.32 | 1051.85 | 32.31 | 951.37 | 25.91 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 612.00 | 0.00 | 52380.00 | 0.00 | 56411.00 | 0.00 |
| **ACSS-1** | 1133.51 | 31.43 | 1065.98 | 25.12 | 8394.45 | 4777.49 | 24938.63 | 10.46 |
| **ACSS-2** | 1141.19 | 34.25 | 1078.92 | 34.64 | 10433.83 | 4535.56 | 24942.79 | 12.53 |
| **ACSS-3** | 1069.39 | 47.39 | 1063.07 | 28.99 | 4875.54 | 3231.67 | 24925.80 | 12.05 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 56331.70 | 0.00 | 31446.80 | 0.00 | 49381.00 | 0.00 |
| **ACSS-1** | 1154.43 | 43.55 | 30927.32 | 45.03 | 43501.18 | 4823.44 | 2356.84 | 3035.08 |
| **ACSS-2** | 2516.86 | 2165.79 | 37846.83 | 4277.41 | 48634.58 | 1713.35 | 25528.03 | 13441.85 |
| **ACSS-3** | 1114.27 | 42.85 | 30897.94 | 24.36 | 34782.39 | 3870.39 | 1198.52 | 36.55 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 487.00 | 0.00 | 62357.00 | 0.00 | 74354.20 | 0.00 |
| **ACSS-1** | 1092.76 | 35.44 | 1148.45 | 14.38 | 28982.98 | 18.35 | 74595.90 | 2034.07 |
| **ACSS-2** | 1109.91 | 29.71 | 1152.56 | 16.24 | 28995.11 | 17.74 | 74876.37 | 2120.22 |
| **ACSS-3** | 1042.06 | 42.19 | 1140.04 | 14.63 | 28961.34 | 23.29 | 72546.15 | 1202.27 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 14439.00 | 0.00 | 20375.00 | 0.00 | 27360.70 | 0.00 |
| **ACSS-1** | 1117.57 | 29.82 | 1127.92 | 24.77 | 20918.18 | 28.61 | 19027.51 | 6.65 |
| **ACSS-2** | 1113.34 | 37.60 | 1130.28 | 25.41 | 20920.37 | 30.83 | 19022.75 | 9.57 |
| **ACSS-3** | 1063.32 | 48.98 | 1099.58 | 33.03 | 20888.07 | 20.03 | 19014.71 | 10.99 |

## Table 64. Dynamic ACSS Solution Cost Results - File 2 (10 Nodes)

| Solution Cost Results - 10 Node ACSS (File 2) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 |
| **ACSS-1** | 944.68 | 17.98 | 943.97 | 12.97 | 943.27 | 17.15 | 944.79 | 16.89 |
| **ACSS-2** | 945.14 | 16.95 | 943.71 | 14.50 | 946.59 | 14.13 | 944.68 | 15.37 |
| **ACSS-3** | 925.32 | 19.28 | 921.77 | 12.67 | 920.76 | 16.73 | 920.41 | 15.15 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 471.25 | 0.00 | 5414.67 | 0.00 | 402.00 | 0.00 |
| **ACSS-1** | 943.30 | 17.91 | 943.30 | 17.91 | 944.31 | 13.51 | 968.64 | 17.86 |
| **ACSS-2** | 951.56 | 13.27 | 944.71 | 13.13 | 967.86 | 24.77 | 917.52 | 11.25 |
| **ACSS-3** | 927.68 | 15.90 | 932.06 | 17.27 | 932.01 | 23.02 | 894.04 | 17.39 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 612.00 | 0.00 | 52380.00 | 0.00 | 56411.00 | 0.00 |
| **ACSS-1** | 954.11 | 20.61 | 934.98 | 20.81 | 1056.69 | 16.81 | 24997.10 | 18.43 |
| **ACSS-2** | 954.83 | 19.00 | 932.73 | 14.34 | 1053.31 | 23.73 | 25221.73 | 798.25 |
| **ACSS-3** | 938.22 | 23.16 | 929.78 | 16.98 | 1037.03 | 22.45 | 24980.16 | 25.32 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 56331.70 | 0.00 | 31446.80 | 0.00 | 49381.00 | 0.00 |
| **ACSS-1** | 952.42 | 10.82 | 1113.88 | 32.01 | 1065.57 | 24.18 | 31478.83 | 1276.56 |
| **ACSS-2** | 946.25 | 12.66 | 1124.65 | 30.94 | 1072.15 | 26.15 | 31811.75 | 2455.74 |
| **ACSS-3** | 939.67 | 13.55 | 1068.96 | 42.76 | 1054.01 | 29.57 | 29316.10 | 2743.42 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 487.00 | 0.00 | 62357.00 | 0.00 | 74354.20 | 0.00 |
| **ACSS-1** | 966.99 | 12.33 | 13277.19 | 939.42 | 14953.71 | 2380.27 | 63735.83 | 4474.59 |
| **ACSS-2** | 964.76 | 19.98 | 13107.01 | 1150.12 | 15076.74 | 1983.58 | 67495.07 | 5114.35 |
| **ACSS-3** | 961.08 | 14.47 | 11450.57 | 1688.63 | 11726.28 | 3043.65 | 58885.56 | 5305.47 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 14439.00 | 0.00 | 20375.00 | 0.00 | 27360.70 | 0.00 |
| **ACSS-1** | 903.79 | 16.09 | 1045.21 | 22.15 | 1143.65 | 26.95 | 39602.20 | 4276.76 |
| **ACSS-2** | 897.41 | 17.77 | 1048.10 | 14.24 | 1215.05 | 373.02 | 42965.75 | 5820.59 |
| **ACSS-3** | 890.00 | 16.01 | 1024.17 | 19.37 | 1128.13 | 31.30 | 39469.10 | 3751.62 |

**Table 65. Dynamic ACSS Solution Cost Results - File 3 (10 Nodes)**

| Solution Cost Results - 10 Node ACSS (File 3) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 372.40 | 0.00 | 372.40 | 0.00 | 372.40 | 0.00 |
| **ACSS-1** | 902.74 | 13.00 | 901.55 | 17.00 | 907.27 | 9.70 | 909.12 | 9.87 |
| **ACSS-2** | 906.88 | 9.23 | 903.64 | 15.84 | 903.79 | 14.30 | 908.10 | 13.60 |
| **ACSS-3** | 887.19 | 17.80 | 889.19 | 21.34 | 890.62 | 17.68 | 892.09 | 11.50 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 339.60 | 0.00 | 350.67 | 0.00 | 345.50 | 0.00 |
| **ACSS-1** | 919.84 | 16.51 | 919.84 | 16.51 | 931.82 | 22.12 | 947.35 | 17.39 |
| **ACSS-2** | 926.39 | 16.16 | 930.97 | 17.85 | 945.27 | 17.85 | 959.32 | 14.42 |
| **ACSS-3** | 904.35 | 16.03 | 904.47 | 27.50 | 927.78 | 14.96 | 938.67 | 10.57 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 383.60 | 0.00 | 30398.00 | 0.00 | 28456.80 | 0.00 |
| **ACSS-1** | 955.04 | 12.70 | 915.00 | 9.87 | 30883.71 | 8.37 | 28859.47 | 9.06 |
| **ACSS-2** | 951.51 | 14.57 | 916.21 | 11.22 | 30888.23 | 8.43 | 28860.47 | 10.06 |
| **ACSS-3** | 943.82 | 20.36 | 904.27 | 15.32 | 30869.63 | 13.53 | 28843.12 | 17.95 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 368.10 | 0.00 | 399.00 | 0.00 | 27353.80 | 0.00 |
| **ACSS-1** | 945.82 | 11.12 | 951.78 | 12.70 | 1006.61 | 15.13 | 27968.29 | 18.98 |
| **ACSS-2** | 943.29 | 10.65 | 951.37 | 12.84 | 1005.06 | 19.11 | 28024.43 | 24.72 |
| **ACSS-3** | 926.97 | 15.10 | 938.49 | 13.28 | 984.83 | 19.87 | 27962.83 | 17.51 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 447.40 | 0.00 | 37420.80 | 0.00 | 107326.00 | 0.00 |
| **ACSS-1** | 1001.37 | 11.18 | 995.41 | 15.57 | 1186.35 | 30.32 | 66234.45 | 1489.22 |
| **ACSS-2** | 999.66 | 11.89 | 997.39 | 14.12 | 1183.62 | 34.19 | 66621.29 | 1469.01 |
| **ACSS-3** | 977.61 | 22.36 | 992.01 | 22.04 | 1175.13 | 31.19 | 65332.39 | 1119.59 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 395.00 | 0.00 | 431.00 | 0.00 | 81322.00 | 0.00 |
| **ACSS-1** | 990.14 | 17.02 | 954.23 | 9.68 | 21054.18 | 24.49 | 90094.01 | 2116.87 |
| **ACSS-2** | 988.05 | 16.46 | 957.90 | 9.37 | 21061.81 | 24.21 | 90689.74 | 2958.18 |
| **ACSS-3** | 969.59 | 22.74 | 953.46 | 12.79 | 21014.65 | 26.21 | 88441.18 | 1965.77 |

**Table 66. Dynamic ACSS Solution Cost Results - File 4 (10 Nodes)**

| Solution Cost Results - 10 Node ACSS (File 4) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 21410.00 | 0.00 | 21410.00 | 0.00 | 21410.00 | 0.00 | 21410.00 | 0.00 |
| ACSS-1 | 976.68 | 14.85 | 976.28 | 18.30 | 978.69 | 16.56 | 978.23 | 16.27 |
| ACSS-2 | 976.40 | 13.21 | 978.77 | 19.82 | 979.13 | 15.90 | 981.82 | 18.22 |
| ACSS-3 | 964.57 | 27.44 | 971.83 | 15.94 | 965.29 | 15.61 | 969.27 | 17.52 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 21410.00 | 0.00 | 517.00 | 0.00 | 416.00 | 0.00 | 459.00 | 0.00 |
| ACSS-1 | 1054.11 | 17.48 | 1054.11 | 17.48 | 1012.56 | 21.91 | 971.53 | 13.67 |
| ACSS-2 | 1055.11 | 14.29 | 1015.01 | 16.18 | 971.67 | 14.65 | 984.49 | 12.09 |
| ACSS-3 | 1030.55 | 25.54 | 1002.38 | 18.87 | 963.10 | 15.55 | 978.14 | 8.08 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 21410.00 | 0.00 | 22416.20 | 0.00 | 25462.00 | 0.00 | 23442.40 | 0.00 |
| ACSS-1 | 998.28 | 26.08 | 1076.81 | 28.29 | 1075.69 | 16.86 | 23909.52 | 38.59 |
| ACSS-2 | 991.90 | 19.25 | 1082.86 | 24.33 | 1074.41 | 16.94 | 23921.76 | 46.12 |
| ACSS-3 | 989.02 | 21.30 | 1061.25 | 28.42 | 1071.60 | 21.08 | 23896.99 | 39.19 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 21410.00 | 0.00 | 554.60 | 0.00 | 36434.40 | 0.00 | 91353.20 | 0.00 |
| ACSS-1 | 1011.25 | 18.53 | 1094.07 | 36.11 | 11388.48 | 932.99 | 69606.76 | 4088.74 |
| ACSS-2 | 1015.30 | 16.28 | 1089.87 | 38.96 | 11485.63 | 793.29 | 68383.80 | 4250.12 |
| ACSS-3 | 1004.49 | 20.25 | 1067.33 | 35.90 | 9474.72 | 3242.95 | 64848.04 | 2957.03 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 21410.00 | 0.00 | 14439.50 | 0.00 | 51358.00 | 0.00 | 38390.00 | 0.00 |
| ACSS-1 | 1057.71 | 13.02 | 13638.22 | 3508.88 | 33739.79 | 909.74 | 47762.74 | 2652.23 |
| ACSS-2 | 1057.61 | 19.21 | 16832.38 | 3306.61 | 33843.16 | 756.43 | 49488.39 | 4082.64 |
| ACSS-3 | 1054.28 | 19.92 | 10187.65 | 4104.00 | 33430.67 | 1209.36 | 45871.50 | 1500.12 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 21410.00 | 0.00 | 26512.40 | 0.00 | 53361.00 | 0.00 | 76328.40 | 0.00 |
| ACSS-1 | 1056.38 | 18.06 | 1050.30 | 11.35 | 68814.41 | 7.26 | 83426.54 | 2814.19 |
| ACSS-2 | 1056.65 | 21.30 | 1047.90 | 10.93 | 68813.99 | 9.96 | 83246.91 | 3343.22 |
| ACSS-3 | 1049.16 | 17.79 | 1037.06 | 16.99 | 68546.61 | 1010.74 | 79231.48 | 4395.94 |

Table 67. Dynamic ACSS Solution Cost Results - File 5 (10 Nodes)

| Solution Cost Results - 10 Node ACSS (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 466.47 | 0.00 | 466.47 | 0.00 | 466.47 | 0.00 |
| ACSS-1 | 968.67 | 14.96 | 973.22 | 15.40 | 966.38 | 19.33 | 970.65 | 16.17 |
| ACSS-2 | 972.75 | 15.35 | 969.38 | 11.46 | 966.12 | 16.90 | 970.77 | 14.66 |
| ACSS-3 | 927.72 | 17.85 | 938.77 | 16.21 | 933.24 | 17.06 | 933.59 | 21.19 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 468.25 | 0.00 | 464.30 | 0.00 | 476.30 | 0.00 |
| ACSS-1 | 1018.70 | 18.81 | 1018.70 | 18.81 | 1031.36 | 25.14 | 991.70 | 16.67 |
| ACSS-2 | 1020.33 | 20.44 | 1032.45 | 29.63 | 994.43 | 17.80 | 975.21 | 14.51 |
| ACSS-3 | 981.24 | 17.57 | 991.59 | 19.78 | 981.60 | 18.34 | 963.33 | 9.48 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 520.25 | 0.00 | 55393.80 | 0.00 | 71366.00 | 0.00 |
| ACSS-1 | 1008.23 | 19.02 | 1118.02 | 14.13 | 11963.23 | 880.37 | 22736.52 | 264.80 |
| ACSS-2 | 1001.68 | 17.25 | 1113.03 | 20.99 | 12201.62 | 13.26 | 22824.67 | 257.46 |
| ACSS-3 | 982.46 | 25.75 | 1100.10 | 24.60 | 11825.99 | 1041.38 | 22734.64 | 260.43 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 474.60 | 0.00 | 15418.70 | 0.00 | 535.00 | 0.00 |
| ACSS-1 | 985.79 | 16.32 | 1038.35 | 24.64 | 1199.82 | 34.44 | 8329.47 | 13.20 |
| ACSS-2 | 984.69 | 21.60 | 1044.21 | 28.25 | 1202.43 | 24.20 | 8336.90 | 15.38 |
| ACSS-3 | 962.75 | 29.30 | 1034.18 | 18.46 | 1177.33 | 23.29 | 8317.18 | 17.10 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 33424.00 | 0.00 | 32391.00 | 0.00 | 58353.00 | 0.00 |
| ACSS-1 | 966.98 | 21.61 | 1067.42 | 14.70 | 32971.63 | 13.35 | 86675.30 | 3.31 |
| ACSS-2 | 974.03 | 14.37 | 1069.30 | 19.38 | 32987.00 | 11.04 | 86675.80 | 1.83 |
| ACSS-3 | 941.83 | 28.98 | 1039.15 | 32.56 | 32935.01 | 24.16 | 86673.37 | 5.28 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 466.47 | 0.00 | 28388.00 | 0.00 | 10434.00 | 0.00 | 77318.70 | 0.00 |
| ACSS-1 | 959.38 | 14.21 | 28972.63 | 28.95 | 59762.57 | 7.02 | 104605.60 | 2.19 |
| ACSS-2 | 958.98 | 15.12 | 28990.09 | 28.00 | 59764.17 | 6.66 | 104604.53 | 3.71 |
| ACSS-3 | 942.68 | 20.37 | 28951.63 | 24.11 | 59756.83 | 4.04 | 104600.80 | 5.77 |

## Table 68. Dynamic ACSS Solution Cost Results - File 6 (10 Nodes)

| Solution Cost Results - 10 Node ACSS (File 6) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 367.00 | 0.00 | 367.00 | 0.00 | 367.00 | 0.00 |
| **ACSS-1** | 940.90 | 12.84 | 944.16 | 14.21 | 942.44 | 17.76 | 943.44 | 17.70 |
| **ACSS-2** | 940.59 | 15.15 | 942.51 | 22.37 | 941.80 | 20.45 | 938.07 | 17.44 |
| **ACSS-3** | 925.06 | 16.44 | 918.83 | 20.56 | 923.38 | 10.30 | 924.43 | 18.24 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 352.00 | 0.00 | 368.00 | 0.00 | 356.50 | 0.00 |
| **ACSS-1** | 973.45 | 17.70 | 973.45 | 17.70 | 904.97 | 10.09 | 902.68 | 11.13 |
| **ACSS-2** | 979.56 | 17.19 | 902.57 | 14.16 | 907.56 | 13.62 | 937.23 | 15.69 |
| **ACSS-3** | 948.60 | 14.55 | 893.00 | 13.69 | 889.99 | 13.22 | 917.08 | 14.82 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 391.50 | 0.00 | 427.75 | 0.00 | 21424.00 | 0.00 |
| **ACSS-1** | 953.70 | 15.67 | 1023.48 | 17.70 | 951.73 | 9.27 | 47820.88 | 25.38 |
| **ACSS-2** | 953.57 | 16.98 | 1023.69 | 20.05 | 953.23 | 10.98 | 47835.66 | 17.87 |
| **ACSS-3** | 935.45 | 17.13 | 997.01 | 24.79 | 937.12 | 37.38 | 47801.55 | 13.89 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 369.50 | 0.00 | 443.40 | 0.00 | 6483.60 | 0.00 |
| **ACSS-1** | 1010.64 | 19.42 | 989.48 | 14.63 | 1127.76 | 19.56 | 1112.56 | 15.57 |
| **ACSS-2** | 2754.96 | 2308.95 | 38914.85 | 3479.90 | 49086.49 | 1955.96 | 25136.84 | 12930.65 |
| **ACSS-3** | 980.27 | 22.97 | 969.66 | 13.87 | 1101.49 | 29.97 | 1104.61 | 17.23 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 402.60 | 0.00 | 442.60 | 0.00 | 121214.00 | 0.00 |
| **ACSS-1** | 1024.16 | 19.75 | 989.20 | 18.35 | 1205.73 | 30.60 | 135510.10 | 2.07 |
| **ACSS-2** | 1024.70 | 19.25 | 994.57 | 13.12 | 1203.05 | 35.97 | 135522.63 | 9.87 |
| **ACSS-3** | 998.89 | 24.32 | 975.10 | 23.04 | 1188.21 | 41.11 | 135510.67 | 3.98 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 396.20 | 0.00 | 21415.00 | 0.00 | 45384.00 | 0.00 |
| **ACSS-1** | 1041.82 | 19.33 | 1141.66 | 25.29 | 1138.77 | 4.90 | 50975.20 | 0.00 |
| **ACSS-2** | 1036.26 | 20.90 | 1138.24 | 24.54 | 1138.07 | 1.92 | 50975.20 | 0.00 |
| **ACSS-3** | 1014.62 | 27.20 | 1087.62 | 47.95 | 1146.65 | 17.98 | 50975.20 | 0.00 |

**Table 69. Dynamic ACSS Solution Cost Results - File 7 (10 Nodes)**

| Solution Cost Results - 10 Node ACSS (File 7) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 369.00 | 0.00 | 369.00 | 0.00 | 369.00 | 0.00 |
| **ACSS-1** | 857.36 | 15.58 | 859.72 | 19.84 | 864.36 | 16.13 | 867.07 | 14.16 |
| **ACSS-2** | 863.34 | 15.29 | 864.06 | 19.69 | 869.07 | 14.71 | 864.57 | 15.64 |
| **ACSS-3** | 857.65 | 16.26 | 853.21 | 16.68 | 849.72 | 15.36 | 857.08 | 15.26 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 350.00 | 0.00 | 393.50 | 0.00 | 371.00 | 0.00 |
| **ACSS-1** | 915.85 | 26.03 | 915.85 | 26.03 | 921.69 | 14.93 | 884.96 | 15.71 |
| **ACSS-2** | 921.81 | 23.90 | 920.33 | 12.06 | 882.94 | 20.32 | 859.02 | 10.92 |
| **ACSS-3** | 906.96 | 29.23 | 899.47 | 14.22 | 865.55 | 16.10 | 855.66 | 14.67 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 27319.00 | 0.00 | 25377.00 | 0.00 | 20415.40 | 0.00 |
| **ACSS-1** | 894.66 | 12.71 | 27773.93 | 10.98 | 25948.40 | 17.84 | 20876.27 | 13.33 |
| **ACSS-2** | 899.57 | 11.74 | 27824.90 | 14.00 | 25950.15 | 16.66 | 20885.35 | 13.60 |
| **ACSS-3** | 886.91 | 15.63 | 27774.33 | 11.53 | 25922.03 | 14.81 | 20858.04 | 12.49 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 366.50 | 0.00 | 452.00 | 0.00 | 410.00 | 0.00 |
| **ACSS-1** | 886.82 | 23.66 | 962.10 | 14.03 | 995.73 | 16.71 | 1115.08 | 33.19 |
| **ACSS-2** | 881.51 | 22.78 | 960.59 | 16.09 | 996.93 | 12.72 | 1114.29 | 31.18 |
| **ACSS-3** | 878.26 | 26.07 | 932.35 | 14.97 | 977.94 | 18.70 | 1082.47 | 23.31 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 395.40 | 0.00 | 492.00 | 0.00 | 486.00 | 0.00 |
| **ACSS-1** | 913.41 | 33.70 | 966.23 | 15.42 | 3978.85 | 1722.61 | 25913.69 | 1901.02 |
| **ACSS-2** | 917.18 | 15.92 | 966.05 | 12.19 | 3712.98 | 1895.45 | 26522.33 | 2267.19 |
| **ACSS-3** | 908.08 | 14.23 | 913.53 | 131.07 | 2823.77 | 1510.37 | 25245.61 | 764.64 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 24362.80 | 0.00 | 26458.00 | 0.00 | 104270.00 | 0.00 |
| **ACSS-1** | 953.32 | 14.89 | 1069.88 | 17.66 | 19992.73 | 11.32 | 156464.23 | 0.63 |
| **ACSS-2** | 943.39 | 19.83 | 1074.14 | 16.46 | 19998.95 | 11.22 | 156463.80 | 1.10 |
| **ACSS-3** | 943.03 | 21.66 | 1058.38 | 22.41 | 19985.49 | 10.14 | 156463.17 | 1.02 |

**Table 70. Dynamic ACSS Solution Cost Results - File 8 (10 Nodes)**

| Solution Cost Results - 10 Node ACSS (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 11409.00 | 0.00 | 11409.00 | 0.00 | 11409.00 | 0.00 |
| **ACSS-1** | 861.35 | 8.34 | 860.90 | 11.46 | 862.60 | 11.97 | 857.70 | 12.16 |
| **ACSS-2** | 860.33 | 12.02 | 859.72 | 15.48 | 857.91 | 13.58 | 861.15 | 9.57 |
| **ACSS-3** | 844.97 | 12.82 | 846.84 | 15.64 | 844.67 | 11.83 | 843.23 | 13.44 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 11371.00 | 0.00 | 37331.40 | 0.00 | 35334.00 | 0.00 |
| **ACSS-1** | 937.35 | 10.75 | 937.35 | 10.75 | 900.22 | 17.49 | 37814.17 | 14.57 |
| **ACSS-2** | 940.40 | 10.03 | 904.64 | 12.68 | 37823.19 | 12.49 | 35786.72 | 8.20 |
| **ACSS-3** | 929.81 | 14.41 | 893.84 | 16.79 | 37806.93 | 11.81 | 35780.50 | 8.89 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 426.00 | 0.00 | 4427.00 | 0.00 | 36367.00 | 0.00 |
| **ACSS-1** | 872.29 | 8.12 | 895.67 | 17.35 | 1180.75 | 555.00 | 27891.48 | 22.48 |
| **ACSS-2** | 873.89 | 12.06 | 892.35 | 20.73 | 1090.76 | 23.00 | 27931.14 | 21.62 |
| **ACSS-3** | 864.93 | 13.57 | 870.13 | 47.72 | 1054.70 | 45.71 | 27880.43 | 19.41 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 420.80 | 0.00 | 13401.00 | 0.00 | 5445.00 | 0.00 |
| **ACSS-1** | 908.22 | 11.49 | 952.95 | 13.62 | 1100.77 | 21.58 | 36521.14 | 1522.39 |
| **ACSS-2** | 909.46 | 12.95 | 953.62 | 11.82 | 1105.95 | 31.90 | 40700.64 | 5347.85 |
| **ACSS-3** | 902.85 | 12.02 | 934.29 | 19.92 | 1093.50 | 24.25 | 35672.91 | 1730.10 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 8418.00 | 0.00 | 454.00 | 0.00 | 30441.00 | 0.00 |
| **ACSS-1** | 891.60 | 10.84 | 982.00 | 7.92 | 1049.86 | 14.55 | 1207.25 | 7.91 |
| **ACSS-2** | 894.13 | 8.78 | 982.20 | 8.57 | 1051.22 | 13.42 | 1208.73 | 3.15 |
| **ACSS-3** | 884.66 | 10.50 | 967.57 | 11.64 | 1032.86 | 8.21 | 1194.28 | 19.37 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 463.00 | 0.00 | 76254.00 | 0.00 | 108289.00 | 0.00 |
| **ACSS-1** | 934.43 | 14.56 | 1076.56 | 31.31 | 31998.12 | 50.28 | 87108.38 | 2081.13 |
| **ACSS-2** | 932.05 | 14.16 | 1078.58 | 26.73 | 32017.84 | 35.03 | 87612.58 | 1314.56 |
| **ACSS-3** | 922.26 | 14.74 | 1063.60 | 38.40 | 31977.27 | 51.00 | 83655.03 | 4106.93 |

## Table 71. Dynamic ACSS Solution Cost Results - File 9 (10 Nodes)

| Solution Cost Results - 10 Node ACSS (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 410.00 | 0.00 | 410.00 | 0.00 | 410.00 | 0.00 | 410.00 | 0.00 |
| ACSS-1 | 924.69 | 23.09 | 921.98 | 38.28 | 928.70 | 18.61 | 934.64 | 15.01 |
| ACSS-2 | 929.26 | 30.99 | 922.30 | 34.53 | 932.47 | 18.16 | 931.25 | 19.53 |
| ACSS-3 | 900.56 | 26.15 | 900.10 | 31.60 | 897.71 | 33.40 | 903.16 | 32.26 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 410.00 | 0.00 | 384.40 | 0.00 | 355.83 | 0.00 | 321.00 | 0.00 |
| ACSS-1 | 879.73 | 30.91 | 879.73 | 30.91 | 895.26 | 15.66 | 904.20 | 14.30 |
| ACSS-2 | 891.99 | 15.33 | 899.23 | 18.92 | 906.27 | 14.85 | 896.43 | 14.98 |
| ACSS-3 | 860.64 | 61.19 | 876.21 | 19.27 | 884.48 | 11.76 | 878.60 | 12.32 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 410.00 | 0.00 | 430.00 | 0.00 | 24529.80 | 0.00 | 22454.80 | 0.00 |
| ACSS-1 | 955.94 | 15.68 | 831.26 | 118.81 | 24974.82 | 85.56 | 22946.43 | 36.28 |
| ACSS-2 | 956.04 | 13.88 | 894.31 | 102.33 | 24994.59 | 93.98 | 22948.21 | 49.12 |
| ACSS-3 | 941.42 | 16.02 | 890.55 | 110.92 | 24897.91 | 108.91 | 22914.42 | 46.69 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 410.00 | 0.00 | 392.75 | 0.00 | 432.00 | 0.00 | 531.80 | 0.00 |
| ACSS-1 | 972.36 | 36.82 | 1019.61 | 21.49 | 1042.86 | 24.60 | 1124.70 | 10.61 |
| ACSS-2 | 987.80 | 18.53 | 1026.69 | 18.56 | 1048.25 | 18.37 | 1125.85 | 10.02 |
| ACSS-3 | 953.05 | 24.32 | 990.87 | 32.67 | 1033.53 | 23.53 | 1114.13 | 29.55 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 410.00 | 0.00 | 426.00 | 0.00 | 547.25 | 0.00 | 89378.00 | 0.00 |
| ACSS-1 | 936.86 | 23.25 | 1043.65 | 17.28 | 1123.33 | 12.45 | 48127.18 | 6006.47 |
| ACSS-2 | 943.69 | 16.09 | 1049.40 | 16.84 | 1115.03 | 18.32 | 50507.82 | 4741.46 |
| ACSS-3 | 933.17 | 16.15 | 1038.18 | 24.15 | 1120.23 | 14.67 | 41276.90 | 8015.04 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 410.00 | 0.00 | 437.00 | 0.00 | 21401.00 | 0.00 | 92267.00 | 0.00 |
| ACSS-1 | 926.67 | 18.85 | 1019.18 | 89.88 | 26948.05 | 15.78 | 110393.97 | 1725.04 |
| ACSS-2 | 932.46 | 19.81 | 1061.99 | 81.73 | 26955.23 | 18.22 | 110292.87 | 2331.20 |
| ACSS-3 | 906.56 | 42.88 | 997.54 | 73.44 | 26920.73 | 23.17 | 109713.73 | 15.39 |

**Table 72. Dynamic ACSS Solution Cost Results - File 10 (10 Nodes)**

| Solution Cost Results - 10 Node ACSS (File 10) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 30328.40 | 0.00 | 30328.40 | 0.00 | 30328.40 | 0.00 |
| **ACSS-1** | 30759.13 | 10.51 | 30757.90 | 12.01 | 30755.75 | 12.11 | 30755.93 | 14.00 |
| **ACSS-2** | 30755.77 | 14.04 | 30754.30 | 15.59 | 30753.23 | 13.47 | 30752.63 | 12.20 |
| **ACSS-3** | 30745.11 | 14.80 | 30744.57 | 13.64 | 30742.03 | 14.28 | 30742.75 | 15.80 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 29334.00 | 0.00 | 26342.60 | 0.00 | 26360.30 | 0.00 |
| **ACSS-1** | 30786.53 | 10.99 | 30786.53 | 10.99 | 29768.20 | 8.98 | 26752.71 | 14.37 |
| **ACSS-2** | 30786.43 | 10.55 | 29767.63 | 11.87 | 26753.79 | 11.01 | 27049.47 | 551.22 |
| **ACSS-3** | 30762.24 | 17.49 | 29753.62 | 15.86 | 26744.98 | 11.65 | 26915.61 | 29.61 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 377.00 | 0.00 | 479.40 | 0.00 | 52338.00 | 0.00 |
| **ACSS-1** | 30759.45 | 10.32 | 855.90 | 11.94 | 975.88 | 12.14 | 21946.23 | 26.73 |
| **ACSS-2** | 30763.75 | 12.89 | 857.16 | 10.38 | 975.51 | 15.44 | 21955.18 | 27.26 |
| **ACSS-3** | 30739.71 | 14.24 | 848.45 | 14.70 | 967.97 | 15.89 | 21922.60 | 33.94 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 382.40 | 0.00 | 388.00 | 0.00 | 30351.20 | 0.00 |
| **ACSS-1** | 30805.18 | 11.26 | 962.70 | 24.73 | 1007.07 | 21.74 | 1021.80 | 17.72 |
| **ACSS-2** | 30810.31 | 10.94 | 964.53 | 19.38 | 1013.48 | 15.26 | 1029.93 | 20.38 |
| **ACSS-3** | 30788.45 | 13.01 | 933.83 | 23.35 | 992.72 | 21.12 | 1013.80 | 18.56 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 35389.00 | 0.00 | 450.00 | 0.00 | 22495.00 | 0.00 |
| **ACSS-1** | 30807.34 | 6.82 | 35880.31 | 17.97 | 1088.07 | 10.13 | 28089.90 | 358.50 |
| **ACSS-2** | 30806.04 | 10.70 | 35884.41 | 16.40 | 1087.87 | 13.60 | 28117.37 | 354.84 |
| **ACSS-3** | 30794.25 | 12.49 | 35839.80 | 34.25 | 1081.77 | 17.85 | 27158.77 | 990.90 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 419.67 | 0.00 | 29429.00 | 0.00 | 63365.60 | 0.00 |
| **ACSS-1** | 30854.64 | 14.67 | 1027.77 | 28.75 | 30115.41 | 29.35 | 91279.36 | 4575.95 |
| **ACSS-2** | 30862.00 | 15.44 | 1032.74 | 18.26 | 30122.32 | 45.99 | 91312.91 | 4957.85 |
| **ACSS-3** | 30837.55 | 14.91 | 1028.85 | 24.49 | 30079.92 | 38.95 | 84853.08 | 3257.22 |

## Appendix D : Phase 2 (Dynamic) Results - 15 Node Network

### Table 73. Dynamic ACSE Solution Cost Results - File 1 (15 Nodes)

| Solution Cost Results - 15 Node ACSE (File 1) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 |
| **ACSE-1** | 7563.19 | 5635.76 | 6155.07 | 4446.69 | 7130.07 | 6960.51 | 7244.33 | 7158.80 |
| **ACSE-2** | 6647.55 | 5782.24 | 6681.52 | 5128.84 | 9324.80 | 6079.47 | 5185.38 | 4209.49 |
| **ACSE-3** | 6944.79 | 4851.08 | 4626.42 | 3797.84 | 6841.35 | 7023.44 | 6926.81 | 6450.46 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 700.63 | 0.00 | 746.60 | 0.00 | 758.00 | 0.00 |
| **ACSE-1** | 7596.07 | 6395.40 | 6195.87 | 5238.69 | 18593.98 | 8075.07 | 16958.62 | 11952.52 |
| **ACSE-2** | 8190.00 | 7318.77 | 5893.36 | 4261.01 | 21131.68 | 8743.58 | 16603.49 | 10020.11 |
| **ACSE-3** | 6641.32 | 4000.26 | 7215.16 | 6393.54 | 15318.82 | 7050.57 | 16632.72 | 10762.98 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 674.50 | 0.00 | 710.33 | 0.00 | 746.20 | 0.00 |
| **ACSE-1** | 9252.84 | 7583.89 | 30781.78 | 12349.53 | 15790.59 | 8448.50 | 19382.94 | 17046.47 |
| **ACSE-2** | 8036.37 | 6078.50 | 28124.32 | 9471.18 | 14076.30 | 3986.60 | 17043.49 | 14875.61 |
| **ACSE-3** | 6168.58 | 5094.89 | 19633.74 | 10463.47 | 16866.77 | 9926.70 | 15619.14 | 14489.47 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 777.50 | 0.00 | 879.20 | 0.00 | 49841.00 | 0.00 |
| **ACSE-1** | 8068.72 | 5704.52 | 16849.11 | 9402.87 | 7651.80 | 50.44 | 16010.16 | 7753.12 |
| **ACSE-2** | 7889.44 | 5357.49 | 16174.91 | 6248.51 | 7650.32 | 50.46 | 18532.45 | 6608.06 |
| **ACSE-3** | 6567.10 | 4724.55 | 14959.33 | 7578.36 | 7357.56 | 2735.58 | 16833.48 | 7748.01 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 39729.80 | 0.00 | 82769.00 | 0.00 | 211614.00 | 0.00 |
| **ACSE-1** | 7078.89 | 4116.82 | 73075.56 | 11484.70 | 123193.23 | 28061.26 | 245120.90 | 9647.67 |
| **ACSE-2** | 8464.97 | 6047.55 | 65293.82 | 14598.28 | 131038.34 | 27211.27 | 245215.50 | 8932.38 |
| **ACSE-3** | 7496.16 | 5327.08 | 68980.89 | 9828.98 | 134440.11 | 23674.69 | 242263.00 | 12692.81 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 796.60 | 0.00 | 1898.00 | 0.00 | 19903.00 | 0.00 |
| **ACSE-1** | 6098.86 | 4957.78 | 16208.96 | 9810.60 | 4626.24 | 3247.95 | 22966.14 | 13076.42 |
| **ACSE-2** | 7547.46 | 6398.61 | 15866.29 | 10350.98 | 4175.93 | 1548.39 | 15753.18 | 13912.85 |
| **ACSE-3** | 5670.21 | 3912.65 | 12744.40 | 10615.44 | 6384.19 | 5905.19 | 25509.31 | 13805.37 |

## Table 74. Dynamic ACSE Solution Cost Results - File 2 (15 Nodes)

| Solution Cost Results - 15 Node ACSE (File 2) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 |
| ACSE-1 | 12953.53 | 8121.47 | 13953.82 | 8400.54 | 16937.00 | 9304.17 | 11480.94 | 6960.77 |
| ACSE-2 | 13486.88 | 7735.17 | 12786.67 | 7769.75 | 13927.73 | 6257.91 | 12104.33 | 8532.26 |
| ACSE-3 | 13359.37 | 8661.15 | 14933.28 | 10448.88 | 10589.64 | 8430.56 | 11361.43 | 9215.11 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 700.63 | 0.00 | 746.60 | 0.00 | 758.00 | 0.00 |
| ACSE-1 | 12244.75 | 6654.67 | 28054.38 | 5090.54 | 25938.88 | 9835.51 | 11841.54 | 4373.51 |
| ACSE-2 | 12575.26 | 7058.42 | 26238.95 | 6722.14 | 23734.92 | 7833.86 | 11292.56 | 4297.52 |
| ACSE-3 | 12197.34 | 7419.82 | 17772.24 | 11186.01 | 18927.77 | 10483.26 | 10103.66 | 6793.58 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 674.50 | 0.00 | 710.33 | 0.00 | 746.20 | 0.00 |
| ACSE-1 | 10959.29 | 7874.03 | 28158.66 | 11844.98 | 9111.15 | 3657.24 | 16288.77 | 8809.58 |
| ACSE-2 | 13222.62 | 7229.57 | 25931.54 | 10514.26 | 12035.72 | 3644.16 | 19954.74 | 11712.02 |
| ACSE-3 | 10227.57 | 6330.54 | 25392.79 | 10971.52 | 15731.39 | 11422.17 | 24565.95 | 12038.67 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 777.50 | 0.00 | 879.20 | 0.00 | 49841.00 | 0.00 |
| ACSE-1 | 13719.62 | 7434.74 | 53912.86 | 6414.68 | 61252.60 | 11954.62 | 82796.16 | 17530.23 |
| ACSE-2 | 11869.96 | 6030.12 | 52417.01 | 4099.67 | 57860.59 | 8717.97 | 70421.36 | 14542.31 |
| ACSE-3 | 10340.85 | 6704.09 | 55345.61 | 8222.44 | 59232.98 | 12169.01 | 76971.09 | 15278.30 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 39729.80 | 0.00 | 82769.00 | 0.00 | 211614.00 | 0.00 |
| ACSE-1 | 10031.34 | 4299.35 | 37198.58 | 12560.93 | 70543.16 | 18457.35 | 123798.36 | 28920.96 |
| ACSE-2 | 13192.15 | 7591.91 | 41937.08 | 18003.42 | 65205.40 | 15053.94 | 148793.07 | 16579.14 |
| ACSE-3 | 12363.50 | 8012.88 | 35689.92 | 21594.47 | 60873.82 | 18567.31 | 133035.33 | 23351.73 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 796.60 | 0.00 | 1898.00 | 0.00 | 19903.00 | 0.00 |
| ACSE-1 | 13051.47 | 7271.74 | 14269.69 | 9329.45 | 76333.34 | 6687.05 | 129021.30 | 8471.82 |
| ACSE-2 | 11378.53 | 5638.13 | 17197.65 | 10325.47 | 75053.92 | 17646.85 | 129404.10 | 14150.68 |
| ACSE-3 | 9234.01 | 5810.77 | 12970.04 | 10291.50 | 77330.80 | 8626.46 | 119537.01 | 17171.70 |

**Table 75. Dynamic ACSE Solution Cost Results - File 3 (15 Nodes)**

| Solution Cost Results - 15 Node ACSE (File 3) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 735.00 | 0.00 | 779.00 | 0.00 | 841.00 | 0.00 |
| **ACSE-1** | 15778.54 | 10592.68 | 15039.00 | 10629.80 | 14397.36 | 9493.36 | 14261.84 | 7532.16 |
| **ACSE-2** | 14125.27 | 11519.48 | 12383.28 | 6532.21 | 17566.43 | 11625.18 | 16927.04 | 11887.85 |
| **ACSE-3** | 16372.23 | 11627.70 | 16348.12 | 12049.50 | 14158.65 | 12445.26 | 16239.67 | 10112.47 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 735.00 | 0.00 | 779.00 | 0.00 | 841.00 | 0.00 |
| **ACSE-1** | 14302.38 | 9745.90 | 39662.91 | 20484.04 | 22564.29 | 11860.00 | 32251.74 | 17264.80 |
| **ACSE-2** | 15399.20 | 11543.38 | 35966.76 | 15911.13 | 23933.82 | 12183.34 | 37383.63 | 20625.80 |
| **ACSE-3** | 14190.73 | 9230.55 | 36847.02 | 17397.79 | 22795.38 | 12438.97 | 41874.82 | 19527.49 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 855.80 | 0.00 | 975.87 | 0.00 | 982.13 | 0.00 |
| **ACSE-1** | 14350.72 | 8675.12 | 46996.29 | 8167.73 | 76959.78 | 12155.94 | 105489.16 | 16988.11 |
| **ACSE-2** | 17037.71 | 10166.59 | 37754.72 | 15370.87 | 77749.35 | 10962.37 | 104644.00 | 16690.32 |
| **ACSE-3** | 20969.79 | 10118.10 | 31266.40 | 12076.66 | 70544.19 | 17075.24 | 101312.41 | 13360.47 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 768.65 | 0.00 | 830.70 | 0.00 | 954.80 | 0.00 |
| **ACSE-1** | 14371.44 | 9659.07 | 14751.66 | 13361.42 | 24688.23 | 17017.56 | 18857.01 | 4787.04 |
| **ACSE-2** | 13713.68 | 7778.06 | 18027.52 | 15484.79 | 26727.57 | 15378.04 | 16428.89 | 4532.15 |
| **ACSE-3** | 15789.72 | 14503.86 | 16127.94 | 6781.01 | 20647.09 | 14858.02 | 18701.03 | 5241.27 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 893.20 | 0.00 | 1127.83 | 0.00 | 115904.00 | 0.00 |
| **ACSE-1** | 15761.07 | 10759.74 | 38944.99 | 15881.22 | 52965.23 | 11875.78 | 133400.87 | 17997.99 |
| **ACSE-2** | 12270.72 | 9421.41 | 33481.73 | 14738.79 | 49071.95 | 11173.92 | 138568.80 | 18260.65 |
| **ACSE-3** | 16283.29 | 10486.46 | 32176.57 | 14974.70 | 52020.68 | 9483.92 | 133191.07 | 19778.48 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 771.00 | 0.00 | 39913.00 | 0.00 | 39023.80 | 0.00 |
| **ACSE-1** | 16441.36 | 9066.17 | 15736.70 | 12101.98 | 61880.44 | 9371.77 | 127159.20 | 19669.29 |
| **ACSE-2** | 16369.70 | 11303.89 | 13430.89 | 10204.33 | 71431.88 | 11080.39 | 138383.67 | 8023.50 |
| **ACSE-3** | 14875.28 | 10807.06 | 17924.68 | 11395.46 | 66157.45 | 13129.03 | 134539.33 | 16736.33 |

## Table 76. Dynamic ACSE Solution Cost Results - File 4 (15 Nodes)

| Solution Cost Results - 15 Node ACSE (File 4) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 42836.00 | 0.00 | 42836.00 | 0.00 | 42836.00 | 0.00 |
| **ACSE-1** | 67126.48 | 8748.93 | 65883.28 | 5118.55 | 63957.40 | 2941.27 | 63912.97 | 3802.32 |
| **ACSE-2** | 65418.15 | 10824.30 | 69487.83 | 12345.15 | 70669.53 | 12963.93 | 67605.20 | 9658.12 |
| **ACSE-3** | 58352.40 | 27330.40 | 67303.83 | 7578.33 | 64738.16 | 7548.74 | 62800.32 | 7298.65 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 52838.50 | 0.00 | 51858.80 | 0.00 | 51833.00 | 0.00 |
| **ACSE-1** | 71239.28 | 12902.71 | 71150.12 | 13886.86 | 63880.92 | 9047.66 | 75019.03 | 9857.15 |
| **ACSE-2** | 66971.77 | 11057.45 | 72238.58 | 19307.79 | 64548.54 | 11019.98 | 73342.81 | 10617.20 |
| **ACSE-3** | 77364.87 | 17614.49 | 65456.49 | 14053.92 | 65730.60 | 9732.90 | 77789.04 | 16369.71 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 42869.30 | 0.00 | 987.53 | 0.00 | 1012.00 | 0.00 |
| **ACSE-1** | 66928.30 | 9066.14 | 58010.48 | 8827.08 | 82116.77 | 18779.37 | 40844.88 | 19245.60 |
| **ACSE-2** | 68267.92 | 11194.68 | 60863.51 | 9553.80 | 85716.06 | 27201.68 | 53943.26 | 22037.47 |
| **ACSE-3** | 71980.92 | 14336.64 | 61387.21 | 10700.38 | 83883.79 | 19608.45 | 41064.89 | 20980.46 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 85712.00 | 0.00 | 76762.40 | 0.00 | 95755.50 | 0.00 |
| **ACSE-1** | 68180.31 | 11982.79 | 114332.88 | 14022.30 | 93695.68 | 9775.61 | 132566.57 | 16944.76 |
| **ACSE-2** | 65640.02 | 4578.36 | 108516.49 | 10555.72 | 91224.92 | 7094.67 | 123928.10 | 1754.41 |
| **ACSE-3** | 66787.99 | 25700.36 | 111202.99 | 12822.42 | 93308.10 | 11229.33 | 136309.53 | 18507.28 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 904.00 | 0.00 | 861.33 | 0.00 | 77844.50 | 0.00 |
| **ACSE-1** | 68210.95 | 7671.15 | 29122.82 | 13249.17 | 15691.27 | 10749.90 | 167378.03 | 26264.18 |
| **ACSE-2** | 71687.01 | 15953.24 | 26869.94 | 14035.26 | 22372.36 | 15897.13 | 163947.43 | 17672.41 |
| **ACSE-3** | 72119.86 | 15239.06 | 32976.64 | 16672.46 | 26714.26 | 12299.00 | 142228.43 | 32877.44 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 817.80 | 0.00 | 46979.70 | 0.00 | 75918.00 | 0.00 |
| **ACSE-1** | 70534.94 | 13953.01 | 10543.46 | 6700.40 | 92201.64 | 14481.90 | 256990.23 | 11886.99 |
| **ACSE-2** | 69972.34 | 15714.33 | 8632.47 | 5258.17 | 86146.57 | 13427.24 | 248693.87 | 15408.62 |
| **ACSE-3** | 63443.81 | 19506.06 | 10496.31 | 4480.07 | 91323.94 | 11806.18 | 251963.77 | 13808.75 |

**Table 77. Dynamic ACSE Solution Cost Results - File 5 (15 Nodes)**

| Solution Cost Results - 15 Node ACSE (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 747.00 | 0.00 | 747.00 | 0.00 | 747.00 | 0.00 |
| **ACSE-1** | 10398.15 | 6869.19 | 7997.69 | 6801.09 | 7544.95 | 5970.70 | 8462.51 | 6236.06 |
| **ACSE-2** | 7716.60 | 6010.57 | 11215.49 | 8041.89 | 8525.19 | 5760.79 | 11637.23 | 7445.84 |
| **ACSE-3** | 8787.05 | 7198.35 | 9831.33 | 15087.29 | 7894.83 | 8479.50 | 7756.26 | 5866.93 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 776.60 | 0.00 | 790.00 | 0.00 | 799.25 | 0.00 |
| **ACSE-1** | 11021.44 | 6840.87 | 13982.41 | 6582.87 | 8174.20 | 6046.66 | 10428.21 | 4576.29 |
| **ACSE-2** | 10262.94 | 8149.77 | 14330.90 | 5865.01 | 8772.47 | 6794.55 | 12579.17 | 4887.23 |
| **ACSE-3** | 12376.42 | 7372.34 | 15181.53 | 9694.79 | 7503.24 | 6678.95 | 15984.40 | 12413.29 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 775.33 | 0.00 | 970.40 | 0.00 | 43834.20 | 0.00 |
| **ACSE-1** | 9607.84 | 6568.05 | 38342.05 | 15971.18 | 25353.92 | 17532.48 | 87961.17 | 20067.34 |
| **ACSE-2** | 8544.64 | 7885.25 | 38251.84 | 16333.39 | 17453.20 | 12919.45 | 80101.75 | 13894.80 |
| **ACSE-3** | 10272.42 | 8780.67 | 30423.59 | 16004.28 | 28362.00 | 15971.27 | 83948.50 | 18346.30 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 749.92 | 0.00 | 764.00 | 0.00 | 794.40 | 0.00 |
| **ACSE-1** | 9815.50 | 6938.85 | 19476.47 | 8329.82 | 77875.00 | 20970.12 | 52519.32 | 16074.90 |
| **ACSE-2** | 9315.70 | 7600.05 | 14871.60 | 10333.16 | 74309.74 | 13005.73 | 51422.73 | 13503.77 |
| **ACSE-3** | 10080.22 | 7132.07 | 17817.19 | 13507.93 | 71056.28 | 15619.62 | 50816.86 | 17066.44 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 768.40 | 0.00 | 832.00 | 0.00 | 1009.88 | 0.00 |
| **ACSE-1** | 8284.05 | 5642.79 | 6221.44 | 4620.04 | 7987.41 | 6254.10 | 98116.01 | 24684.53 |
| **ACSE-2** | 8462.68 | 6399.38 | 9875.92 | 6433.43 | 6437.22 | 5557.29 | 115951.26 | 23912.10 |
| **ACSE-3** | 10279.50 | 9011.56 | 6822.22 | 3550.55 | 7869.52 | 5924.43 | 101897.59 | 23229.55 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 765.50 | 0.00 | 49890.00 | 0.00 | 126821.00 | 0.00 |
| **ACSE-1** | 9618.81 | 7648.58 | 41781.91 | 6668.14 | 102003.82 | 21007.36 | 247081.67 | 7278.71 |
| **ACSE-2** | 8782.81 | 4882.91 | 45186.38 | 7577.29 | 93381.77 | 24370.96 | 243312.03 | 11503.26 |
| **ACSE-3** | 11247.54 | 6208.61 | 35957.76 | 10892.55 | 95916.49 | 27290.45 | 250688.43 | 7539.91 |

**Table 78. Dynamic ACSE Solution Cost Results - File 6 (15 Nodes)**

| Solution Cost Results - 15 Node ACSE (File 6) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 811.25 | 0.00 | 811.25 | 0.00 | 811.25 | 0.00 | 811.25 | 0.00 |
| ACSE-1 | 19917.23 | 11395.04 | 18563.33 | 9494.64 | 20473.42 | 10848.52 | 14599.93 | 8949.50 |
| ACSE-2 | 16032.70 | 10280.35 | 19251.75 | 11416.08 | 16154.63 | 10812.28 | 18608.66 | 10302.93 |
| ACSE-3 | 19928.97 | 13770.08 | 19222.19 | 11009.17 | 17635.05 | 12503.82 | 22863.18 | 12265.39 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 811.25 | 0.00 | 879.00 | 0.00 | 1022.02 | 0.00 | 1107.87 | 0.00 |
| ACSE-1 | 20580.38 | 11269.83 | 32477.68 | 8106.34 | 24724.47 | 14518.02 | 54136.65 | 19425.90 |
| ACSE-2 | 18700.42 | 13061.86 | 32672.20 | 7729.85 | 28812.18 | 16046.42 | 64169.30 | 15401.43 |
| ACSE-3 | 16719.33 | 10682.56 | 27843.76 | 12112.40 | 27074.82 | 15961.28 | 62074.62 | 19589.94 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 811.25 | 0.00 | 853.15 | 0.00 | 55158.60 | 0.00 | 50920.20 | 0.00 |
| ACSE-1 | 21719.46 | 13620.66 | 34432.69 | 6883.02 | 56170.22 | 12947.34 | 48601.49 | 11610.61 |
| ACSE-2 | 21484.09 | 9500.64 | 31830.82 | 9702.68 | 61859.82 | 13366.37 | 50582.58 | 9098.17 |
| ACSE-3 | 21243.38 | 11968.74 | 26270.43 | 11783.57 | 52204.25 | 18240.06 | 50553.56 | 15183.52 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 811.25 | 0.00 | 772.00 | 0.00 | 13766.80 | 0.00 | 13790.00 | 0.00 |
| ACSE-1 | 18317.39 | 7725.59 | 16852.66 | 9917.37 | 27021.39 | 6186.33 | 50805.64 | 10156.76 |
| ACSE-2 | 22167.71 | 12114.31 | 22499.32 | 15506.09 | 28703.89 | 8396.53 | 54738.29 | 11329.81 |
| ACSE-3 | 21370.67 | 12605.10 | 15748.61 | 10215.32 | 27140.13 | 8063.91 | 52464.84 | 12729.19 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 811.25 | 0.00 | 898.00 | 0.00 | 888.50 | 0.00 | 2960.33 | 0.00 |
| ACSE-1 | 15245.86 | 10128.11 | 35528.47 | 10940.98 | 23356.32 | 9868.08 | 25895.44 | 21928.39 |
| ACSE-2 | 17720.22 | 10642.19 | 35140.07 | 11814.40 | 29118.21 | 12570.84 | 24418.09 | 20508.45 |
| ACSE-3 | 19904.85 | 12274.63 | 38302.36 | 17717.89 | 24038.38 | 8193.87 | 24431.99 | 17694.09 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 811.25 | 0.00 | 833.00 | 0.00 | 908.35 | 0.00 | 133843.00 | 0.00 |
| ACSE-1 | 20183.11 | 11372.22 | 24910.55 | 14480.11 | 25496.04 | 9667.99 | 209375.67 | 51460.44 |
| ACSE-2 | 23135.53 | 14722.30 | 27323.63 | 13017.62 | 29864.09 | 9413.76 | 212033.10 | 44759.21 |
| ACSE-3 | 21190.87 | 13366.30 | 23388.21 | 10459.00 | 24730.04 | 10389.43 | 222300.93 | 58599.65 |

## Table 79. Dynamic ACSE Solution Cost Results - File 7 (15 Nodes)

| Solution Cost Results - 15 Node ACSE (File 7) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 829.80 | 0.00 | 829.80 | 0.00 | 829.80 | 0.00 |
| **ACSE-1** | 14782.30 | 6581.82 | 12302.89 | 7095.56 | 10564.00 | 6303.07 | 10285.69 | 6575.71 |
| **ACSE-2** | 10186.61 | 3633.30 | 14106.85 | 6146.37 | 13362.21 | 7496.22 | 11063.60 | 4704.73 |
| **ACSE-3** | 14167.40 | 7692.11 | 12103.20 | 6502.41 | 12640.75 | 7765.16 | 13520.43 | 9248.09 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 856.00 | 0.00 | 834.40 | 0.00 | 818.90 | 0.00 |
| **ACSE-1** | 12318.24 | 5224.28 | 18264.69 | 12354.19 | 16731.56 | 13466.90 | 19321.82 | 9075.72 |
| **ACSE-2** | 12925.68 | 6011.26 | 14871.18 | 11606.25 | 17439.84 | 11753.04 | 23997.33 | 11814.35 |
| **ACSE-3** | 13545.05 | 7358.50 | 14526.89 | 9452.91 | 10418.39 | 9267.20 | 18023.11 | 10424.32 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 838.00 | 0.00 | 1168.85 | 0.00 | 1084.87 | 0.00 |
| **ACSE-1** | 11633.17 | 4882.45 | 22362.94 | 11298.19 | 37889.33 | 14557.27 | 37417.39 | 8318.56 |
| **ACSE-2** | 13255.78 | 5475.99 | 23472.64 | 11628.66 | 36660.48 | 16466.89 | 36809.85 | 6122.51 |
| **ACSE-3** | 12323.37 | 9073.39 | 26265.50 | 11515.95 | 35000.12 | 12652.71 | 42851.73 | 12582.52 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 784.00 | 0.00 | 901.00 | 0.00 | 936.20 | 0.00 |
| **ACSE-1** | 12751.89 | 5320.43 | 24096.94 | 16064.62 | 103081.59 | 20951.59 | 71603.56 | 19065.04 |
| **ACSE-2** | 13134.91 | 6167.92 | 22706.82 | 13655.94 | 97440.12 | 16488.82 | 77965.33 | 21578.66 |
| **ACSE-3** | 11378.70 | 5395.62 | 26408.52 | 17627.08 | 98492.95 | 21630.00 | 81241.25 | 26927.30 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 864.67 | 0.00 | 895.00 | 0.00 | 52112.20 | 0.00 |
| **ACSE-1** | 12213.67 | 7147.40 | 27318.91 | 13986.61 | 39808.69 | 16046.32 | 31964.97 | 7506.40 |
| **ACSE-2** | 12877.09 | 7137.05 | 32260.81 | 13226.45 | 47846.57 | 12654.76 | 33651.23 | 6578.26 |
| **ACSE-3** | 10723.31 | 6130.68 | 24438.97 | 12184.77 | 37506.88 | 16588.34 | 41887.30 | 16611.48 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 900.80 | 0.00 | 807.93 | 0.00 | 985.75 | 0.00 |
| **ACSE-1** | 13786.25 | 9185.03 | 55976.35 | 25023.75 | 31675.40 | 15661.86 | 29514.85 | 9750.40 |
| **ACSE-2** | 10988.22 | 5052.91 | 51944.36 | 20003.82 | 31629.29 | 15061.90 | 29139.15 | 4472.78 |
| **ACSE-3** | 10410.90 | 6284.73 | 45396.65 | 19893.26 | 36574.09 | 17558.09 | 25405.47 | 7897.70 |

**Table 80. Dynamic ACSE Solution Cost Results - File 8 (15 Nodes)**

| Solution Cost Results - 15 Node ACSE (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 788.25 | 0.00 | 788.25 | 0.00 | 788.25 | 0.00 |
| ACSE-1 | 23971.33 | 14571.72 | 20967.75 | 12846.38 | 20174.70 | 11340.81 | 24114.15 | 16412.02 |
| ACSE-2 | 22109.23 | 15689.62 | 22874.33 | 13487.04 | 24452.79 | 17312.77 | 32360.48 | 26445.82 |
| ACSE-3 | 20367.16 | 10929.90 | 19268.38 | 21732.93 | 16660.22 | 15451.29 | 22613.51 | 21165.26 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 840.75 | 0.00 | 933.70 | 0.00 | 925.17 | 0.00 |
| ACSE-1 | 25906.66 | 14855.13 | 20783.01 | 11236.87 | 23325.01 | 16169.38 | 13049.30 | 2711.82 |
| ACSE-2 | 26644.59 | 18996.39 | 20738.30 | 12998.83 | 24977.34 | 14543.52 | 13701.10 | 6169.43 |
| ACSE-3 | 27037.62 | 17858.77 | 15451.98 | 10609.44 | 19594.43 | 10591.74 | 15096.64 | 9660.22 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 825.00 | 0.00 | 48864.50 | 0.00 | 77794.00 | 0.00 |
| ACSE-1 | 24247.09 | 15337.95 | 40237.48 | 20976.77 | 76548.57 | 16574.11 | 85529.11 | 7674.81 |
| ACSE-2 | 22768.83 | 17227.28 | 36355.06 | 19290.89 | 67779.55 | 13381.21 | 87941.34 | 10348.71 |
| ACSE-3 | 30991.97 | 19467.00 | 45600.20 | 22255.96 | 75061.62 | 15653.35 | 87303.34 | 14566.80 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 836.25 | 0.00 | 44774.60 | 0.00 | 45835.40 | 0.00 |
| ACSE-1 | 28657.60 | 18912.60 | 9658.58 | 6674.78 | 73708.49 | 10779.36 | 93404.27 | 7044.95 |
| ACSE-2 | 27260.00 | 22708.18 | 9950.86 | 9252.65 | 69895.11 | 11607.64 | 83845.14 | 18449.93 |
| ACSE-3 | 25483.80 | 18864.31 | 11925.23 | 8085.14 | 72341.86 | 16440.64 | 87134.98 | 13970.72 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 764.45 | 0.00 | 833.20 | 0.00 | 58801.20 | 0.00 |
| ACSE-1 | 25700.08 | 20065.73 | 16394.40 | 6584.08 | 47192.83 | 15367.02 | 99271.21 | 12546.05 |
| ACSE-2 | 28545.17 | 21912.01 | 15328.08 | 10599.42 | 58046.77 | 20192.54 | 92790.48 | 3794.54 |
| ACSE-3 | 26960.37 | 15096.23 | 12817.49 | 8112.41 | 50620.93 | 16573.95 | 91754.58 | 2148.72 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 925.20 | 0.00 | 15849.10 | 0.00 | 1928.50 | 0.00 |
| ACSE-1 | 22560.67 | 15316.53 | 31742.83 | 16413.70 | 54242.66 | 17512.47 | 137776.83 | 6761.58 |
| ACSE-2 | 24430.65 | 16190.86 | 35038.01 | 18447.31 | 59117.22 | 12760.85 | 134234.53 | 1653.77 |
| ACSE-3 | 24197.46 | 20979.59 | 29117.06 | 16695.15 | 61529.10 | 27072.10 | 137546.10 | 6860.30 |

**Table 81. Dynamic ACSE Solution Cost Results - File 9 (15 Nodes)**

| Solution Cost Results - 15 Node ACSE (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 730.60 | 0.00 | 730.60 | 0.00 | 730.60 | 0.00 |
| **ACSE-1** | 6603.80 | 4751.72 | 7565.54 | 5733.73 | 6471.85 | 5628.77 | 6553.95 | 5261.63 |
| **ACSE-2** | 5699.27 | 4040.82 | 8096.09 | 11396.65 | 7398.12 | 10588.77 | 5795.19 | 4119.78 |
| **ACSE-3** | 5690.75 | 5077.49 | 7396.91 | 5405.43 | 7108.18 | 4873.08 | 10251.02 | 9973.19 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 816.00 | 0.00 | 865.25 | 0.00 | 856.75 | 0.00 |
| **ACSE-1** | 6072.38 | 6079.43 | 22755.92 | 23605.78 | 58136.02 | 8119.93 | 57781.53 | 14497.05 |
| **ACSE-2** | 5587.75 | 4416.26 | 36979.45 | 23204.48 | 60245.11 | 8615.61 | 54167.44 | 17345.13 |
| **ACSE-3** | 9679.56 | 8994.69 | 34189.56 | 20699.30 | 59105.06 | 10862.65 | 54737.66 | 16598.75 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 755.67 | 0.00 | 741.00 | 0.00 | 2845.90 | 0.00 |
| **ACSE-1** | 4790.08 | 3160.31 | 7746.66 | 4825.11 | 9425.01 | 6907.42 | 31786.59 | 12258.85 |
| **ACSE-2** | 6660.34 | 5257.94 | 7081.36 | 4293.99 | 10450.72 | 5896.46 | 34688.64 | 10155.48 |
| **ACSE-3** | 7559.61 | 5561.88 | 7836.54 | 6422.82 | 9416.50 | 6669.94 | 29507.27 | 16198.46 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 748.92 | 0.00 | 806.20 | 0.00 | 868.00 | 0.00 |
| **ACSE-1** | 5516.80 | 4753.90 | 28478.55 | 14900.21 | 32127.92 | 11199.01 | 58325.84 | 14714.73 |
| **ACSE-2** | 5960.43 | 3722.02 | 31667.10 | 12453.71 | 37084.26 | 10812.81 | 55426.45 | 19301.21 |
| **ACSE-3** | 6358.38 | 5178.80 | 24042.75 | 13579.83 | 34383.54 | 11962.25 | 67000.69 | 19256.19 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 818.25 | 0.00 | 816.20 | 0.00 | 24820.10 | 0.00 |
| **ACSE-1** | 7778.09 | 9083.33 | 8087.44 | 5141.83 | 9116.87 | 11471.88 | 56375.45 | 15503.11 |
| **ACSE-2** | 5697.51 | 4811.50 | 12194.67 | 12023.59 | 9441.67 | 12355.90 | 47400.32 | 4884.34 |
| **ACSE-3** | 6375.33 | 5538.95 | 16258.83 | 13374.55 | 12840.17 | 14199.81 | 49072.86 | 7887.35 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 851.00 | 0.00 | 950.30 | 0.00 | 117848.00 | 0.00 |
| **ACSE-1** | 5375.93 | 4806.03 | 21070.40 | 7651.59 | 47018.20 | 10031.17 | 222940.37 | 25683.02 |
| **ACSE-2** | 6072.29 | 4334.72 | 23064.81 | 20550.56 | 52736.89 | 9909.79 | 238455.37 | 31394.97 |
| **ACSE-3** | 6515.72 | 5375.92 | 20782.96 | 8986.54 | 55593.52 | 10439.88 | 226874.90 | 33189.45 |

## Table 82. Dynamic ACSE Solution Cost Results - File 10 (15 Nodes)

| Solution Cost Results - 15 Node ACSE (File 10) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 745.00 | 0.00 | 745.00 | 0.00 | 745.00 | 0.00 |
| **ACSE-1** | 8596.36 | 6340.77 | 11129.51 | 7600.48 | 9391.67 | 6637.81 | 7369.58 | 5511.20 |
| **ACSE-2** | 6382.38 | 6127.49 | 9665.21 | 9121.94 | 7558.45 | 7680.99 | 8355.55 | 5592.22 |
| **ACSE-3** | 8013.78 | 6064.77 | 7336.45 | 5240.22 | 9649.64 | 7228.78 | 9830.21 | 8775.98 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 755.80 | 0.00 | 773.60 | 0.00 | 762.05 | 0.00 |
| **ACSE-1** | 6964.47 | 5896.54 | 14830.59 | 6700.83 | 19504.07 | 11151.35 | 7494.17 | 4674.30 |
| **ACSE-2** | 8758.66 | 7507.70 | 19089.00 | 11590.52 | 23127.62 | 16481.62 | 7934.45 | 5747.98 |
| **ACSE-3** | 9387.94 | 7025.33 | 15653.04 | 10115.85 | 16272.91 | 11298.68 | 12314.39 | 8119.44 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 750.00 | 0.00 | 752.50 | 0.00 | 736.20 | 0.00 |
| **ACSE-1** | 10205.49 | 6535.13 | 7311.52 | 5504.97 | 5985.04 | 4128.52 | 13081.44 | 9679.75 |
| **ACSE-2** | 11069.99 | 8457.51 | 9907.06 | 5430.59 | 6614.16 | 5361.53 | 12939.03 | 12093.51 |
| **ACSE-3** | 7360.88 | 6106.93 | 7521.14 | 5730.63 | 6017.87 | 4382.49 | 13289.30 | 8691.82 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 813.70 | 0.00 | 43833.70 | 0.00 | 31976.20 | 0.00 |
| **ACSE-1** | 9167.54 | 5305.75 | 35742.69 | 9946.83 | 47235.37 | 6634.34 | 81114.39 | 18284.71 |
| **ACSE-2** | 9555.07 | 8866.85 | 33231.84 | 6615.90 | 47784.28 | 6609.06 | 79255.61 | 15946.22 |
| **ACSE-3** | 9936.09 | 7396.94 | 27584.25 | 13999.66 | 49074.45 | 8698.28 | 79768.56 | 14826.96 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 76713.40 | 0.00 | 42806.70 | 0.00 | 20917.00 | 0.00 |
| **ACSE-1** | 8649.49 | 7673.97 | 59143.29 | 12346.83 | 82868.30 | 12410.10 | 67246.30 | 19406.03 |
| **ACSE-2** | 13266.64 | 13079.34 | 57320.34 | 9931.11 | 82965.08 | 10213.55 | 66849.19 | 11350.76 |
| **ACSE-3** | 10918.07 | 9566.40 | 57595.04 | 12002.48 | 95924.99 | 21180.95 | 71912.24 | 20454.60 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 886.20 | 0.00 | 83848.90 | 0.00 | 10029.20 | 0.00 |
| **ACSE-1** | 8340.78 | 6620.15 | 47981.84 | 21398.99 | 120104.19 | 19604.74 | 126063.14 | 26570.48 |
| **ACSE-2** | 8711.84 | 5978.70 | 59171.16 | 20278.36 | 119935.75 | 18453.43 | 106540.74 | 28413.77 |
| **ACSE-3** | 8918.44 | 7260.82 | 47263.31 | 21342.17 | 122601.11 | 16790.87 | 131532.04 | 24295.57 |

**Table 83. Dynamic ACSS Solution Cost Results - File 1 (15 Nodes)**

| Solution Cost Results - 15 Node ACSS (File 1) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 |
| **ACSS-1** | 2389.84 | 52.37 | 2394.65 | 44.00 | 2389.58 | 10.98 | 2360.02 | 65.71 |
| **ACSS-2** | 2370.17 | 41.25 | 2415.30 | 49.17 | 2430.26 | 39.37 | 2373.02 | 62.86 |
| **ACSS-3** | 2284.04 | 27.96 | 2285.62 | 33.39 | 2284.64 | 46.65 | 2322.39 | 19.70 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 700.63 | 0.00 | 746.60 | 0.00 | 758.00 | 0.00 |
| **ACSS-1** | 2522.90 | 35.79 | 2453.43 | 72.35 | 2616.54 | 56.82 | 2528.27 | 176.81 |
| **ACSS-2** | 2485.96 | 50.66 | 2488.96 | 51.71 | 2638.18 | 84.90 | 2600.40 | 58.30 |
| **ACSS-3** | 2382.70 | 59.81 | 2345.92 | 81.05 | 2518.25 | 27.81 | 2517.96 | 38.00 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **ACSS-1** | 2450.87 | 31.75 | 2428.09 | 25.54 | 2456.07 | 20.22 | 2616.12 | 37.72 |
| **ACSS-2** | 2447.58 | 54.09 | 2398.35 | 36.44 | 2496.07 | 32.84 | 2632.07 | 57.35 |
| **ACSS-3** | 2302.03 | 81.79 | 2300.66 | 58.95 | 2438.80 | 69.21 | 2531.54 | 28.72 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 777.50 | 0.00 | 879.20 | 0.00 | 49841.00 | 0.00 |
| **ACSS-1** | 2452.11 | 39.33 | 2524.74 | 21.14 | 2554.44 | 7.54 | 3233.28 | 990.27 |
| **ACSS-2** | 2448.08 | 64.91 | 2517.33 | 35.72 | 2556.11 | 31.80 | 3717.45 | 1650.55 |
| **ACSS-3** | 2315.12 | 39.51 | 2448.65 | 64.16 | 2520.84 | 18.53 | 2830.21 | 87.24 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 39729.80 | 0.00 | 82769.00 | 0.00 | 211614.00 | 0.00 |
| **ACSS-1** | 2505.88 | 43.52 | 41378.29 | 52.78 | 87664.75 | 4775.97 | 211770.30 | 1454.28 |
| **ACSS-2** | 2562.30 | 33.30 | 41462.06 | 115.01 | 89344.54 | 2597.06 | 214130.40 | 886.03 |
| **ACSS-3** | 2410.55 | 72.97 | 41344.54 | 61.79 | 85032.50 | 5041.52 | 210373.60 | 2294.01 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 796.60 | 0.00 | 1898.00 | 0.00 | 19903.00 | 0.00 |
| **ACSS-1** | 2483.71 | 27.49 | 2534.76 | 90.75 | 2554.64 | 89.99 | 7226.63 | 543.05 |
| **ACSS-2** | 2454.06 | 23.22 | 2558.90 | 45.79 | 2557.30 | 60.50 | 3908.85 | 1526.85 |
| **ACSS-3** | 2267.80 | 126.89 | 2467.68 | 59.75 | 2578.15 | 14.85 | 3511.69 | 1259.93 |

**Table 84. Dynamic ACSS Solution Cost Results - File 2 (15 Nodes)**

| Solution Cost Results - 15 Node ACSS (File 2) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 |
| ACSS-1 | 2285.93 | 93.73 | 2381.87 | 116.65 | 2339.55 | 101.31 | 2393.51 | 51.39 |
| ACSS-2 | 2322.26 | 132.18 | 2377.32 | 108.66 | 2356.91 | 121.83 | 2485.35 | 79.99 |
| ACSS-3 | 2307.55 | 126.25 | 2324.43 | 134.99 | 2369.00 | 144.41 | 2276.15 | 162.83 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 700.63 | 0.00 | 746.60 | 0.00 | 758.00 | 0.00 |
| ACSS-1 | 2446.48 | 105.59 | 2527.28 | 75.86 | 2705.15 | 87.03 | 2581.31 | 37.66 |
| ACSS-2 | 2420.46 | 69.93 | 2502.70 | 46.96 | 2716.26 | 51.94 | 2587.17 | 30.27 |
| ACSS-3 | 2381.14 | 116.34 | 2432.34 | 51.16 | 2603.87 | 102.42 | 2524.67 | 29.61 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| ACSS-1 | 2382.85 | 89.95 | 2657.79 | 47.34 | 2509.36 | 36.87 | 2523.68 | 83.75 |
| ACSS-2 | 2345.22 | 95.78 | 2685.16 | 48.33 | 2497.38 | 57.79 | 2555.08 | 58.67 |
| ACSS-3 | 2353.10 | 90.07 | 2553.52 | 70.13 | 2479.64 | 35.79 | 2469.81 | 105.38 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 777.50 | 0.00 | 879.20 | 0.00 | 49841.00 | 0.00 |
| ACSS-1 | 2530.86 | 53.72 | 47440.19 | 57.62 | 52450.83 | 30.96 | 49601.56 | 69.02 |
| ACSS-2 | 2454.96 | 90.34 | 47478.34 | 20.71 | 52476.73 | 31.80 | 49584.14 | 115.67 |
| ACSS-3 | 2419.45 | 74.24 | 47451.35 | 25.11 | 52372.32 | 33.14 | 49480.94 | 116.59 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 39729.80 | 0.00 | 82769.00 | 0.00 | 211614.00 | 0.00 |
| ACSS-1 | 2428.74 | 99.92 | 2613.97 | 50.85 | 6948.12 | 4922.16 | 77367.13 | 8448.45 |
| ACSS-2 | 2418.11 | 113.89 | 2595.80 | 51.34 | 11377.57 | 1942.18 | 74762.72 | 11981.30 |
| ACSS-3 | 2449.18 | 99.23 | 2484.26 | 64.28 | 6293.61 | 3483.13 | 76334.77 | 8748.61 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 796.60 | 0.00 | 1898.00 | 0.00 | 19903.00 | 0.00 |
| ACSS-1 | 2387.88 | 225.16 | 2692.02 | 114.16 | 59116.83 | 1331.19 | 78656.56 | 3891.61 |
| ACSS-2 | 2455.92 | 62.89 | 2690.22 | 159.40 | 59579.75 | 1343.82 | 83810.41 | 904.97 |
| ACSS-3 | 2496.04 | 81.96 | 2740.08 | 43.26 | 58599.34 | 1192.32 | 79441.80 | 1769.41 |

**Table 85. Dynamic ACSS Solution Cost Results - File 3 (15 Nodes)**

| Solution Cost Results - 15 Node ACSS (File 3) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 724.50 | 0.00 | 735.00 | 0.00 | 779.00 | 0.00 | 841.00 | 0.00 |
| ACSS-1 | 2305.52 | 30.75 | 2314.34 | 27.67 | 2301.71 | 46.90 | 2281.89 | 25.71 |
| ACSS-2 | 2283.11 | 39.67 | 2271.04 | 19.29 | 2311.05 | 39.76 | 2246.58 | 84.56 |
| ACSS-3 | 2264.98 | 25.68 | 2275.36 | 30.82 | 2301.33 | 50.98 | 2271.94 | 48.09 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 724.50 | 0.00 | 735.00 | 0.00 | 779.00 | 0.00 | 841.00 | 0.00 |
| ACSS-1 | 2427.64 | 25.93 | 2513.33 | 71.72 | 2730.84 | 123.48 | 4052.57 | 2168.25 |
| ACSS-2 | 2404.36 | 31.97 | 3153.84 | 1263.28 | 2681.94 | 131.91 | 3757.77 | 2084.36 |
| ACSS-3 | 2392.75 | 34.13 | 2472.41 | 54.45 | 2698.42 | 102.23 | 3701.94 | 1494.17 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 724.50 | 0.00 | 855.80 | 0.00 | 975.87 | 0.00 | 982.13 | 0.00 |
| ACSS-1 | 2319.99 | 32.75 | 2424.07 | 46.71 | 8414.98 | 3585.09 | 23441.55 | 6050.80 |
| ACSS-2 | 2307.82 | 24.77 | 2450.60 | 30.81 | 11325.15 | 5149.47 | 26838.55 | 6187.48 |
| ACSS-3 | 2293.18 | 45.33 | 2398.15 | 59.75 | 2517.10 | 62.00 | 18902.76 | 3933.90 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 724.50 | 0.00 | 768.65 | 0.00 | 830.70 | 0.00 | 954.80 | 0.00 |
| ACSS-1 | 2416.29 | 21.28 | 2718.60 | 35.73 | 2522.90 | 65.14 | 4288.97 | 2396.61 |
| ACSS-2 | 2380.30 | 38.07 | 2714.16 | 48.46 | 2422.31 | 118.19 | 4464.19 | 2811.80 |
| ACSS-3 | 2347.81 | 39.11 | 2644.30 | 95.52 | 2453.84 | 80.23 | 2866.99 | 63.19 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 724.50 | 0.00 | 893.20 | 0.00 | 1127.83 | 0.00 | 115904.00 | 0.00 |
| ACSS-1 | 2442.82 | 12.04 | 3317.59 | 1203.29 | 19917.16 | 13223.09 | 90656.41 | 2412.68 |
| ACSS-2 | 2400.36 | 9.16 | 2762.53 | 85.44 | 23153.10 | 4013.07 | 89031.16 | 9186.59 |
| ACSS-3 | 2416.36 | 48.46 | 2757.16 | 83.84 | 17588.50 | 5964.54 | 92081.74 | 3262.51 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 724.50 | 0.00 | 771.00 | 0.00 | 39913.00 | 0.00 | 39023.80 | 0.00 |
| ACSS-1 | 2455.80 | 14.10 | 2630.75 | 78.84 | 41457.77 | 16.48 | 76595.88 | 3342.26 |
| ACSS-2 | 2429.40 | 29.00 | 2673.07 | 43.46 | 41456.44 | 48.82 | 78565.81 | 6156.59 |
| ACSS-3 | 2365.85 | 24.47 | 2568.00 | 70.70 | 41430.78 | 55.98 | 74072.20 | 4659.09 |

## Table 86. Dynamic ACSS Solution Cost Results - File 4 (15 Nodes)

| Solution Cost Results - 15 Node ACSS (File 4) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 42836.00 | 0.00 | 42836.00 | 0.00 | 42836.00 | 0.00 |
| **ACSS-1** | 47295.83 | 4839.59 | 44474.21 | 69.09 | 44832.96 | 1088.94 | 44326.80 | 85.17 |
| **ACSS-2** | 44426.17 | 70.22 | 44360.68 | 61.10 | 44491.05 | 103.24 | 44459.89 | 47.79 |
| **ACSS-3** | 44309.86 | 125.27 | 44347.33 | 66.37 | 44396.00 | 72.46 | 44343.77 | 60.53 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 52838.50 | 0.00 | 51858.80 | 0.00 | 51833.00 | 0.00 |
| **ACSS-1** | 44402.71 | 67.98 | 54392.03 | 49.34 | 53307.96 | 47.10 | 55669.89 | 2283.71 |
| **ACSS-2** | 44732.98 | 715.79 | 54418.52 | 34.95 | 53295.98 | 37.29 | 55523.44 | 2108.98 |
| **ACSS-3** | 44358.82 | 38.42 | 54370.58 | 43.33 | 53252.49 | 44.51 | 53416.95 | 46.84 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 42869.30 | 0.00 | 987.53 | 0.00 | 1012.00 | 0.00 |
| **ACSS-1** | 44457.68 | 92.50 | 44468.14 | 37.77 | 14909.19 | 7286.84 | 2695.21 | 43.54 |
| **ACSS-2** | 44867.72 | 851.89 | 44472.72 | 54.20 | 18863.49 | 4334.09 | 3175.91 | 1273.84 |
| **ACSS-3** | 44350.16 | 60.51 | 44403.24 | 25.66 | 9388.33 | 5781.91 | 2598.33 | 54.95 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 85712.00 | 0.00 | 76762.40 | 0.00 | 95755.50 | 0.00 |
| **ACSS-1** | 44549.49 | 104.47 | 88889.49 | 2176.47 | 78067.24 | 29.29 | 117208.10 | 1556.61 |
| **ACSS-2** | 44519.41 | 86.67 | 87726.46 | 959.31 | 78081.97 | 23.45 | 118130.00 | 35.43 |
| **ACSS-3** | 44280.49 | 36.86 | 87262.08 | 124.09 | 78030.00 | 20.42 | 116742.20 | 2016.60 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 904.00 | 0.00 | 861.33 | 0.00 | 77844.50 | 0.00 |
| **ACSS-1** | 44471.76 | 118.31 | 2997.21 | 977.36 | 2753.40 | 49.26 | 55216.55 | 6522.96 |
| **ACSS-2** | 44449.24 | 23.14 | 2709.33 | 55.56 | 2793.86 | 44.65 | 63714.64 | 4981.00 |
| **ACSS-3** | 44384.36 | 57.35 | 2640.26 | 59.91 | 2687.33 | 57.95 | 60024.69 | 2306.59 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 817.80 | 0.00 | 46979.70 | 0.00 | 75918.00 | 0.00 |
| **ACSS-1** | 44479.74 | 106.38 | 2421.26 | 90.65 | 50991.47 | 3483.28 | 210823.10 | 746.19 |
| **ACSS-2** | 44463.66 | 79.04 | 2433.56 | 38.39 | 49180.44 | 2170.04 | 208217.00 | 6462.64 |
| **ACSS-3** | 44359.14 | 37.16 | 2390.08 | 47.82 | 48728.72 | 1976.85 | 197549.60 | 8365.09 |

Table 87. Dynamic ACSS Solution Cost Results - File 5 (15 Nodes)

| Solution Cost Results - 15 Node ACSS (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0.0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 747.00 | 0.00 | 747.00 | 0.00 | 747.00 | 0.00 | 747.00 | 0.00 |
| ACSS-1 | 2403.59 | 28.79 | 2348.06 | 30.95 | 2332.93 | 50.70 | 2393.49 | 33.98 |
| ACSS-2 | 2340.22 | 46.50 | 2384.94 | 22.42 | 2368.22 | 34.54 | 2356.77 | 33.51 |
| ACSS-3 | 2265.16 | 24.38 | 2261.06 | 23.27 | 2267.87 | 20.99 | 2309.71 | 25.85 |
| 10.0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 747.00 | 0.00 | 776.60 | 0.00 | 790.00 | 0.00 | 799.25 | 0.00 |
| ACSS-1 | 2383.63 | 67.91 | 2546.50 | 47.46 | 2356.59 | 24.88 | 2510.31 | 45.08 |
| ACSS-2 | 2370.10 | 52.90 | 2519.17 | 52.95 | 2357.28 | 49.90 | 2474.20 | 28.09 |
| ACSS-3 | 2305.89 | 17.03 | 2448.40 | 46.13 | 2295.93 | 39.49 | 2372.15 | 65.73 |
| 20.0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 747.00 | 0.00 | 775.33 | 0.00 | 970.40 | 0.00 | 43834.20 | 0.00 |
| ACSS-1 | 2446.62 | 40.68 | 2740.10 | 54.56 | 2825.84 | 43.90 | 48567.24 | 2863.93 |
| ACSS-2 | 2480.28 | 32.92 | 2737.79 | 147.96 | 2737.90 | 242.47 | 46783.08 | 2007.45 |
| ACSS-3 | 2355.02 | 64.46 | 2708.92 | 96.48 | 2843.38 | 81.59 | 46272.19 | 1567.68 |
| 30.0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 747.00 | 0.00 | 749.92 | 0.00 | 764.00 | 0.00 | 794.40 | 0.00 |
| ACSS-1 | 2453.07 | 24.86 | 2550.56 | 32.40 | 9478.96 | 5174.84 | 8608.61 | 6776.42 |
| ACSS-2 | 2459.60 | 35.83 | 2527.05 | 28.55 | 6753.52 | 4078.68 | 12765.79 | 4211.76 |
| ACSS-3 | 2377.62 | 45.91 | 2442.95 | 31.90 | 7452.25 | 3422.67 | 7155.85 | 4751.11 |
| 40.0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 747.00 | 0.00 | 768.40 | 0.00 | 832.00 | 0.00 | 1009.88 | 0.00 |
| ACSS-1 | 2433.74 | 43.15 | 2388.81 | 21.79 | 2574.91 | 30.10 | 52150.11 | 3459.46 |
| ACSS-2 | 2446.30 | 33.95 | 2354.11 | 48.41 | 2567.23 | 38.18 | 46923.84 | 6882.13 |
| ACSS-3 | 2368.87 | 49.54 | 2342.31 | 40.65 | 2568.61 | 19.61 | 50871.65 | 6217.52 |
| 50.0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 747.00 | 0.00 | 765.50 | 0.00 | 49890.00 | 0.00 | 126821.00 | 0.00 |
| ACSS-1 | 2515.68 | 40.93 | 3519.10 | 1723.54 | 51766.70 | 60.50 | 157849.90 | 6254.61 |
| ACSS-2 | 2501.98 | 37.20 | 4732.74 | 2519.46 | 53349.66 | 3323.97 | 161978.80 | 1861.64 |
| ACSS-3 | 2429.88 | 78.33 | 2654.07 | 105.35 | 51640.31 | 69.70 | 158465.90 | 7619.74 |

**Table 88. Dynamic ACSS Solution Cost Results - File 6 (15 Nodes)**

| Solution Cost Results - 15 Node ACSS (File 6) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 811.25 | 0.00 | 811.25 | 0.00 | 811.25 | 0.00 |
| **ACSS-1** | 2481.45 | 56.62 | 2410.66 | 62.51 | 2460.75 | 34.88 | 2396.85 | 55.21 |
| **ACSS-2** | 2454.23 | 49.14 | 2471.65 | 10.24 | 2462.55 | 42.25 | 2476.97 | 27.62 |
| **ACSS-3** | 2481.11 | 33.76 | 2120.75 | 389.15 | 2428.18 | 36.75 | 2386.27 | 55.98 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 879.00 | 0.00 | 1022.02 | 0.00 | 1107.87 | 0.00 |
| **ACSS-1** | 2543.91 | 45.14 | 2702.94 | 30.97 | 2691.23 | 35.64 | 6904.83 | 2111.28 |
| **ACSS-2** | 2554.60 | 54.69 | 3195.76 | 1130.69 | 2706.33 | 51.77 | 7691.31 | 2221.82 |
| **ACSS-3** | 2503.46 | 35.52 | 2645.58 | 54.96 | 2647.12 | 71.84 | 6090.58 | 1520.92 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 853.15 | 0.00 | 55158.60 | 0.00 | 50920.20 | 0.00 |
| **ACSS-1** | 2534.70 | 44.98 | 2731.68 | 40.53 | 6693.24 | 3262.02 | 16268.10 | 4812.58 |
| **ACSS-2** | 2489.41 | 44.33 | 2763.93 | 88.68 | 6299.51 | 4084.37 | 12564.18 | 7052.18 |
| **ACSS-3** | 2485.12 | 43.51 | 2652.16 | 100.69 | 6599.32 | 2378.61 | 13484.58 | 6051.66 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 772.00 | 0.00 | 13766.80 | 0.00 | 13790.00 | 0.00 |
| **ACSS-1** | 2539.09 | 90.03 | 2589.86 | 84.41 | 4543.73 | 2126.76 | 15376.95 | 4500.15 |
| **ACSS-2** | 2548.39 | 31.82 | 2647.90 | 75.43 | 3640.68 | 2052.35 | 16180.54 | 4051.18 |
| **ACSS-3** | 2525.29 | 65.16 | 2494.48 | 67.98 | 3154.88 | 1531.05 | 9953.58 | 4362.26 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 898.00 | 0.00 | 888.50 | 0.00 | 2960.33 | 0.00 |
| **ACSS-1** | 2613.08 | 27.57 | 3379.85 | 1516.77 | 2866.25 | 48.04 | 2833.43 | 69.20 |
| **ACSS-2** | 2569.56 | 33.24 | 2896.26 | 79.84 | 2871.24 | 101.76 | 2831.61 | 76.74 |
| **ACSS-3** | 2608.83 | 57.67 | 2762.47 | 66.46 | 2901.84 | 126.69 | 2811.21 | 44.94 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 833.00 | 0.00 | 908.35 | 0.00 | 133843.00 | 0.00 |
| **ACSS-1** | 2585.04 | 76.30 | 3157.64 | 1361.00 | 2650.35 | 95.50 | 143599.90 | 1865.43 |
| **ACSS-2** | 2543.12 | 46.16 | 2540.29 | 51.30 | 2818.59 | 72.43 | 143010.00 | 0.00 |
| **ACSS-3** | 2489.06 | 71.57 | 2554.06 | 36.82 | 2650.50 | 78.17 | 134124.40 | 12268.60 |

**Table 89. Dynamic ACSS Solution Cost Results - File 7 (15 Nodes)**

| Solution Cost Results - 15 Node ACSS (File 7) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 829.80 | 0.00 | 829.80 | 0.00 | 829.80 | 0.00 | 829.80 | 0.00 |
| ACSS-1 | 2372.41 | 82.40 | 2407.56 | 42.40 | 2410.62 | 64.05 | 2436.07 | 51.59 |
| ACSS-2 | 2404.32 | 35.19 | 2435.66 | 20.38 | 2522.62 | 41.04 | 2420.42 | 25.86 |
| ACSS-3 | 2296.20 | 88.79 | 2389.29 | 49.39 | 2360.93 | 35.93 | 2407.06 | 79.29 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 829.80 | 0.00 | 856.00 | 0.00 | 834.40 | 0.00 | 818.90 | 0.00 |
| ACSS-1 | 2410.20 | 60.25 | 2320.43 | 84.00 | 2500.01 | 23.54 | 2549.86 | 157.59 |
| ACSS-2 | 2465.32 | 31.93 | 2386.50 | 104.99 | 2490.26 | 44.62 | 2568.77 | 57.36 |
| ACSS-3 | 2297.74 | 145.35 | 2378.16 | 94.84 | 2444.46 | 73.92 | 2541.49 | 99.30 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 829.80 | 0.00 | 838.00 | 0.00 | 1168.85 | 0.00 | 1084.87 | 0.00 |
| ACSS-1 | 2514.41 | 64.48 | 2708.83 | 84.07 | 6204.32 | 2287.90 | 3911.22 | 1303.54 |
| ACSS-2 | 2591.92 | 33.89 | 2670.83 | 31.88 | 5568.37 | 2272.72 | 6363.94 | 1204.18 |
| ACSS-3 | 2437.08 | 73.62 | 2727.63 | 116.94 | 5544.28 | 3484.73 | 3798.34 | 1727.15 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 829.80 | 0.00 | 784.00 | 0.00 | 901.00 | 0.00 | 936.20 | 0.00 |
| ACSS-1 | 2508.87 | 75.83 | 2571.03 | 43.75 | 19284.48 | 6466.30 | 8521.82 | 3935.29 |
| ACSS-2 | 2561.23 | 61.22 | 2579.60 | 38.35 | 29146.84 | 10727.11 | 9994.77 | 5173.92 |
| ACSS-3 | 2199.52 | 437.68 | 2533.12 | 44.07 | 19971.78 | 9026.29 | 9303.78 | 3774.48 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 829.80 | 0.00 | 864.67 | 0.00 | 895.00 | 0.00 | 52112.20 | 0.00 |
| ACSS-1 | 2467.63 | 111.95 | 2770.01 | 42.79 | 3368.54 | 1357.57 | 9017.07 | 3413.98 |
| ACSS-2 | 2449.88 | 94.13 | 2792.39 | 29.39 | 4361.25 | 2222.04 | 5666.64 | 2227.86 |
| ACSS-3 | 2349.13 | 129.41 | 2738.22 | 31.85 | 2705.83 | 158.07 | 6006.24 | 3340.05 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 829.80 | 0.00 | 900.80 | 0.00 | 807.93 | 0.00 | 985.75 | 0.00 |
| ACSS-1 | 2488.74 | 32.69 | 4785.07 | 2046.97 | 2697.30 | 45.06 | 3974.39 | 2104.44 |
| ACSS-2 | 2535.11 | 69.54 | 7961.04 | 2741.97 | 2829.05 | 128.15 | 3778.81 | 1673.88 |
| ACSS-3 | 2424.97 | 67.38 | 6476.16 | 2775.46 | 2736.65 | 111.72 | 3432.83 | 1259.83 |

**Table 90. Dynamic ACSS Solution Cost Results - File 8 (15 Nodes)**

| Solution Cost Results - 15 Node ACSS (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 788.25 | 0.00 | 788.25 | 0.00 | 788.25 | 0.00 |
| **ACSS-1** | 2605.05 | 39.18 | 2593.83 | 68.86 | 2536.91 | 56.97 | 2547.96 | 48.18 |
| **ACSS-2** | 2578.09 | 27.64 | 2581.62 | 34.32 | 2557.05 | 78.17 | 2539.48 | 68.11 |
| **ACSS-3** | 2458.67 | 97.84 | 2504.46 | 65.71 | 2427.07 | 51.73 | 2544.41 | 35.58 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 840.75 | 0.00 | 933.70 | 0.00 | 925.17 | 0.00 |
| **ACSS-1** | 2474.35 | 423.84 | 2626.25 | 31.75 | 2938.83 | 83.17 | 2742.02 | 75.11 |
| **ACSS-2** | 2565.48 | 96.02 | 2642.93 | 65.98 | 2845.58 | 87.45 | 2791.15 | 99.23 |
| **ACSS-3** | 2570.93 | 72.91 | 2614.36 | 30.94 | 2815.80 | 65.66 | 2621.98 | 70.26 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 825.00 | 0.00 | 48864.50 | 0.00 | 77794.00 | 0.00 |
| **ACSS-1** | 2440.38 | 277.48 | 5838.29 | 3958.12 | 50594.73 | 55.79 | 53064.28 | 1952.47 |
| **ACSS-2** | 2527.41 | 54.96 | 5300.39 | 2345.81 | 50637.85 | 81.64 | 51421.58 | 235.18 |
| **ACSS-3** | 2437.06 | 123.08 | 3875.92 | 1267.77 | 50529.84 | 41.51 | 53071.96 | 2028.00 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 836.25 | 0.00 | 44774.60 | 0.00 | 45835.40 | 0.00 |
| **ACSS-1** | 2671.23 | 38.42 | 2592.32 | 50.65 | 46514.65 | 118.58 | 49153.19 | 2940.99 |
| **ACSS-2** | 2335.92 | 436.90 | 2580.80 | 40.18 | 47290.55 | 1742.83 | 49369.57 | 2703.58 |
| **ACSS-3** | 2595.56 | 102.70 | 2469.31 | 113.87 | 46452.39 | 100.35 | 48546.00 | 1617.16 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 764.45 | 0.00 | 833.20 | 0.00 | 58801.20 | 0.00 |
| **ACSS-1** | 2663.68 | 18.05 | 2441.63 | 24.78 | 5458.88 | 2251.40 | 90243.95 | 19.57 |
| **ACSS-2** | 2606.64 | 147.46 | 2481.11 | 8.28 | 5816.88 | 2573.14 | 90261.50 | 30.90 |
| **ACSS-3** | 2482.87 | 157.81 | 2405.98 | 63.99 | 7569.81 | 3953.75 | 90214.37 | 44.88 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 925.20 | 0.00 | 15849.10 | 0.00 | 1928.50 | 0.00 |
| **ACSS-1** | 2602.86 | 55.92 | 2848.57 | 115.36 | 7714.49 | 3005.44 | 105319.28 | 12572.03 |
| **ACSS-2** | 2633.02 | 45.02 | 2931.30 | 83.01 | 6500.61 | 3950.83 | 112989.20 | 5938.04 |
| **ACSS-3** | 2576.43 | 42.91 | 3603.95 | 1895.96 | 7018.48 | 4047.39 | 92822.08 | 7796.53 |

**Table 91. Dynamic ACSS Solution Cost Results - File 9 (15 Nodes)**

| Solution Cost Results - 15 Node ACSS (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 730.60 | 0.00 | 730.60 | 0.00 | 730.60 | 0.00 | 730.60 | 0.00 |
| ACSS-1 | 2194.22 | 15.84 | 2192.19 | 5.91 | 2196.19 | 24.39 | 2140.77 | 79.01 |
| ACSS-2 | 2207.39 | 25.32 | 2201.14 | 27.53 | 2174.81 | 10.01 | 2181.69 | 13.12 |
| ACSS-3 | 2181.51 | 17.35 | 2130.49 | 14.48 | 2089.31 | 49.34 | 2167.34 | 25.50 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 730.60 | 0.00 | 816.00 | 0.00 | 865.25 | 0.00 | 856.75 | 0.00 |
| ACSS-1 | 2230.88 | 28.85 | 2259.30 | 125.98 | 3842.79 | 2209.53 | 2752.98 | 24.25 |
| ACSS-2 | 2230.44 | 38.55 | 2343.16 | 75.83 | 10524.38 | 6175.22 | 3594.64 | 1794.02 |
| ACSS-3 | 2181.05 | 22.67 | 2387.88 | 102.05 | 2853.52 | 1732.83 | 2661.87 | 140.39 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 730.60 | 0.00 | 755.67 | 0.00 | 741.00 | 0.00 | 2845.90 | 0.00 |
| ACSS-1 | 2042.81 | 127.63 | 2295.90 | 16.90 | 2350.13 | 36.19 | 2573.33 | 98.57 |
| ACSS-2 | 2188.46 | 25.92 | 2341.93 | 32.60 | 2366.12 | 15.91 | 3073.38 | 956.66 |
| ACSS-3 | 2141.79 | 18.13 | 2321.67 | 55.15 | 2309.76 | 48.91 | 2563.26 | 94.75 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 730.60 | 0.00 | 748.92 | 0.00 | 806.20 | 0.00 | 868.00 | 0.00 |
| ACSS-1 | 2233.15 | 101.50 | 3086.54 | 1242.05 | 8091.70 | 3388.01 | 15019.50 | 10242.03 |
| ACSS-2 | 2310.42 | 23.82 | 2437.65 | 108.72 | 5715.60 | 2561.03 | 11938.84 | 5766.07 |
| ACSS-3 | 2261.10 | 32.76 | 2340.22 | 123.61 | 6303.55 | 4095.26 | 10519.27 | 4123.59 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 730.60 | 0.00 | 818.25 | 0.00 | 816.20 | 0.00 | 24820.10 | 0.00 |
| ACSS-1 | 2282.92 | 35.12 | 2515.12 | 26.57 | 2349.53 | 16.49 | 17071.81 | 2338.72 |
| ACSS-2 | 2243.14 | 71.60 | 2546.00 | 23.19 | 2373.40 | 21.24 | 17403.60 | 3011.18 |
| ACSS-3 | 2212.40 | 45.51 | 2485.97 | 48.61 | 2354.34 | 23.28 | 14142.62 | 2049.26 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 730.60 | 0.00 | 851.00 | 0.00 | 950.30 | 0.00 | 117848.00 | 0.00 |
| ACSS-1 | 2304.73 | 34.36 | 2674.97 | 73.02 | 20753.21 | 4932.46 | 156219.40 | 4245.57 |
| ACSS-2 | 2311.24 | 20.79 | 2610.42 | 58.50 | 20528.66 | 3455.38 | 158214.40 | 10385.20 |
| ACSS-3 | 2251.73 | 35.92 | 2590.36 | 47.74 | 17382.65 | 4793.28 | 155387.70 | 3821.49 |

**Table 92. Dynamic ACSS Solution Cost Results - File 10 (15 Nodes)**

| Solution Cost Results - 15 Node ACSS (File 10) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 745.00 | 0.00 | 745.00 | 0.00 | 745.00 | 0.00 |
| **ACSS-1** | 2345.45 | 16.20 | 2325.61 | 19.72 | 2301.32 | 17.39 | 2362.99 | 34.83 |
| **ACSS-2** | 2355.95 | 13.38 | 2351.92 | 13.47 | 2332.84 | 26.46 | 2351.48 | 35.63 |
| **ACSS-3** | 2214.75 | 27.78 | 2240.18 | 27.88 | 2247.91 | 28.03 | 2270.03 | 11.45 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 755.80 | 0.00 | 773.60 | 0.00 | 762.05 | 0.00 |
| **ACSS-1** | 2419.39 | 36.41 | 2444.17 | 31.03 | 2404.95 | 47.13 | 2387.72 | 213.80 |
| **ACSS-2** | 2407.47 | 35.96 | 2427.77 | 64.43 | 2454.30 | 54.60 | 2364.97 | 146.11 |
| **ACSS-3** | 2315.55 | 21.95 | 2380.13 | 60.31 | 2357.49 | 29.75 | 2337.89 | 95.77 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 750.00 | 0.00 | 752.50 | 0.00 | 736.20 | 0.00 |
| **ACSS-1** | 2396.05 | 22.17 | 2360.20 | 61.81 | 2362.79 | 18.88 | 2317.92 | 28.52 |
| **ACSS-2** | 2403.60 | 20.55 | 2436.71 | 40.62 | 2346.49 | 15.80 | 2360.59 | 16.71 |
| **ACSS-3** | 2355.49 | 48.70 | 2380.92 | 62.66 | 2327.44 | 29.55 | 2309.86 | 65.58 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 813.70 | 0.00 | 43833.70 | 0.00 | 31976.20 | 0.00 |
| **ACSS-1** | 2422.57 | 25.00 | 5311.71 | 2089.31 | 29847.11 | 7265.93 | 47669.55 | 2552.98 |
| **ACSS-2** | 2427.50 | 44.66 | 7231.03 | 350.87 | 25984.23 | 9753.05 | 47713.60 | 1409.15 |
| **ACSS-3** | 2343.79 | 13.00 | 3890.85 | 2100.78 | 21267.73 | 10122.63 | 47394.20 | 4046.96 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 76713.40 | 0.00 | 42806.70 | 0.00 | 20917.00 | 0.00 |
| **ACSS-1** | 2504.95 | 42.68 | 41399.26 | 49.19 | 65434.47 | 11901.88 | 31827.73 | 4244.61 |
| **ACSS-2** | 2526.50 | 26.38 | 41413.56 | 61.89 | 68548.46 | 10395.02 | 35790.20 | 6248.52 |
| **ACSS-3** | 2370.60 | 31.97 | 41336.43 | 26.16 | 63537.60 | 10765.57 | 28117.99 | 2765.33 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 886.20 | 0.00 | 83848.90 | 0.00 | 10029.20 | 0.00 |
| **ACSS-1** | 2430.16 | 55.24 | 2836.14 | 148.49 | 84200.65 | 1494.32 | 53321.54 | 3375.19 |
| **ACSS-2** | 2503.21 | 35.22 | 4595.70 | 3762.46 | 84230.12 | 1568.27 | 52430.78 | 2014.98 |
| **ACSS-3** | 2421.80 | 52.08 | 2724.15 | 56.71 | 83638.97 | 1531.42 | 54255.87 | 4306.03 |

# Appendix E: Phase 3 (Distributed) Results - 10 Node Network

## Table 93. Dynamic DACSE Solution Cost Results - File 1 (10 Nodes)

| Solution Cost Results - 10 Node DACSE (File 1) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 |
| **DACSE-1** | 6479.70 | 1632.28 | 6780.52 | 2267.47 | 6480.84 | 1632.07 | 6183.18 | 6.60 |
| **DACSE-2** | 6778.50 | 2269.52 | 6480.90 | 1635.26 | 6181.31 | 5.33 | 6479.86 | 1635.46 |
| **DACSE-3** | 6182.53 | 5.89 | 6411.85 | 3412.49 | 6911.81 | 2931.44 | 6777.89 | 2270.50 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 471.25 | 0.00 | 5414.67 | 0.00 | 402.00 | 0.00 |
| **DACSE-1** | 6182.48 | 4.14 | 1219.93 | 26.53 | 1193.36 | 5.65 | 2101.41 | 2030.14 |
| **DACSE-2** | 6480.40 | 1636.49 | 1203.73 | 25.47 | 3808.70 | 4971.67 | 9755.72 | 10468.77 |
| **DACSE-3** | 6181.77 | 3.85 | 1209.67 | 28.27 | 1197.14 | 10.59 | 2241.84 | 2098.72 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 612.00 | 0.00 | 52380.00 | 0.00 | 56411.00 | 0.00 |
| **DACSE-1** | 6547.33 | 1638.02 | 24959.96 | 1065.56 | 44577.00 | 1372.78 | 24984.33 | 13.40 |
| **DACSE-2** | 6547.33 | 1638.02 | 24620.44 | 970.89 | 44585.57 | 1376.22 | 25096.41 | 546.52 |
| **DACSE-3** | 6546.87 | 1633.01 | 23989.15 | 4429.73 | 43783.42 | 1904.05 | 24989.42 | 8.19 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 56331.70 | 0.00 | 31446.80 | 0.00 | 49381.00 | 0.00 |
| **DACSE-1** | 6280.72 | 7.59 | 44893.45 | 3437.80 | 48207.38 | 1267.57 | 48955.45 | 13051.16 |
| **DACSE-2** | 6278.12 | 7.38 | 41144.14 | 3997.13 | 48001.19 | 34.11 | 20605.64 | 9056.63 |
| **DACSE-3** | 6577.11 | 1631.64 | 43245.95 | 3198.25 | 48584.82 | 1556.75 | 45333.55 | 15468.23 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 487.00 | 0.00 | 62357.00 | 0.00 | 74354.20 | 0.00 |
| **DACSE-1** | 6205.61 | 8.69 | 7961.05 | 5558.08 | 46315.93 | 3009.76 | 76797.33 | 8.55 |
| **DACSE-2** | 6205.71 | 9.66 | 12838.13 | 6341.95 | 44344.40 | 4377.97 | 76775.87 | 6.24 |
| **DACSE-3** | 6503.30 | 1630.29 | 7254.89 | 6030.92 | 46619.52 | 2557.13 | 76795.03 | 7.31 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 14439.00 | 0.00 | 20375.00 | 0.00 | 27360.70 | 0.00 |
| **DACSE-1** | 6194.06 | 10.09 | 1331.13 | 41.63 | 30968.10 | 0.00 | 40854.00 | 14455.15 |
| **DACSE-2** | 6495.29 | 1645.98 | 1294.57 | 22.04 | 31134.60 | 911.96 | 37768.33 | 15464.39 |
| **DACSE-3** | 6193.36 | 10.07 | 1323.93 | 33.03 | 31300.49 | 1264.94 | 36640.63 | 15549.47 |

## Table 94. Dynamic DACSE Solution Cost Results - File 2 (10 Nodes)

| Solution Cost Results - 10 Node DACSE (File 2) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 |
| **DACSE-1** | 2422.17 | 1808.63 | 3768.85 | 2184.72 | 3235.02 | 1821.64 | 3597.39 | 1599.54 |
| **DACSE-2** | 3148.04 | 2501.39 | 2446.27 | 2186.46 | 2317.73 | 1659.78 | 2317.54 | 1999.18 |
| **DACSE-3** | 2320.45 | 2449.38 | 4058.01 | 2252.19 | 2262.73 | 1866.80 | 2860.47 | 2021.48 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 471.25 | 0.00 | 5414.67 | 0.00 | 402.00 | 0.00 |
| **DACSE-1** | 2945.30 | 1919.79 | 9174.78 | 9305.59 | 7252.61 | 3288.50 | 9840.39 | 1493.74 |
| **DACSE-2** | 2483.63 | 1861.01 | 6368.74 | 7094.74 | 10576.09 | 4714.91 | 11220.73 | 3604.67 |
| **DACSE-3** | 2023.49 | 1616.17 | 5898.99 | 5906.03 | 8235.71 | 3610.88 | 10361.11 | 4643.88 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 612.00 | 0.00 | 52380.00 | 0.00 | 56411.00 | 0.00 |
| **DACSE-1** | 2776.62 | 1827.07 | 1445.85 | 843.51 | 7722.20 | 3643.16 | 41736.25 | 3315.59 |
| **DACSE-2** | 2317.93 | 1760.21 | 1595.57 | 1061.88 | 7494.61 | 5664.15 | 46968.53 | 4712.33 |
| **DACSE-3** | 3197.48 | 2019.64 | 1944.34 | 2069.38 | 8775.92 | 4280.41 | 41978.77 | 3808.24 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 56331.70 | 0.00 | 31446.80 | 0.00 | 49381.00 | 0.00 |
| **DACSE-1** | 2840.62 | 2129.10 | 8539.08 | 4385.62 | 4946.29 | 3043.84 | 43375.10 | 1637.37 |
| **DACSE-2** | 2501.51 | 1796.16 | 7931.29 | 4176.16 | 4114.38 | 6322.80 | 44821.90 | 924.84 |
| **DACSE-3** | 3096.22 | 2782.85 | 10021.03 | 4536.22 | 7044.30 | 7941.87 | 44046.69 | 1769.74 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 487.00 | 0.00 | 62357.00 | 0.00 | 74354.20 | 0.00 |
| **DACSE-1** | 2377.70 | 1814.43 | 22861.69 | 2993.12 | 26164.50 | 0.00 | 73790.40 | 17.18 |
| **DACSE-2** | 3109.24 | 2527.16 | 21344.04 | 2330.84 | 26164.50 | 0.00 | 73813.87 | 25.12 |
| **DACSE-3** | 2831.87 | 2007.27 | 22002.35 | 3611.42 | 26164.50 | 0.00 | 73801.53 | 26.89 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 14439.00 | 0.00 | 20375.00 | 0.00 | 27360.70 | 0.00 |
| **DACSE-1** | 2801.38 | 2151.02 | 9144.67 | 4787.70 | 26947.68 | 3578.63 | 51021.50 | 0.00 |
| **DACSE-2** | 2477.34 | 1893.98 | 9640.26 | 4000.25 | 36391.93 | 7876.34 | 51021.50 | 0.00 |
| **DACSE-3** | 2267.24 | 2432.59 | 10307.18 | 4975.32 | 27612.68 | 4703.24 | 51021.50 | 0.00 |

Table 95. Dynamic DACSE Solution Cost Results - File 3 (10 Nodes)

| Solution Cost Results - 10 Node DACSE (File 3) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 372.40 | 0.00 | 372.40 | 0.00 | 372.40 | 0.00 |
| **DACSE-1** | 4050.83 | 6120.54 | 1035.73 | 17.31 | 1207.57 | 917.94 | 1044.53 | 11.13 |
| **DACSE-2** | 2043.81 | 3816.73 | 3047.90 | 5201.81 | 3548.37 | 5703.31 | 3549.20 | 5702.93 |
| **DACSE-3** | 2046.90 | 3815.87 | 1029.88 | 27.84 | 1198.99 | 919.99 | 1318.06 | 1155.16 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 339.60 | 0.00 | 350.67 | 0.00 | 345.50 | 0.00 |
| **DACSE-1** | 2053.77 | 3816.84 | 1110.40 | 53.20 | 1084.33 | 31.63 | 1081.53 | 5.44 |
| **DACSE-2** | 2552.46 | 4591.43 | 1433.74 | 1137.47 | 1101.13 | 35.76 | 1090.33 | 14.56 |
| **DACSE-3** | 3064.06 | 4078.20 | 1099.87 | 56.07 | 1386.27 | 1640.26 | 1076.53 | 16.55 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 383.60 | 0.00 | 30398.00 | 0.00 | 28456.80 | 0.00 |
| **DACSE-1** | 3627.53 | 5702.97 | 1048.09 | 13.79 | 34261.20 | 2380.72 | 28893.77 | 8.50 |
| **DACSE-2** | 3624.81 | 5704.23 | 1054.24 | 16.59 | 35596.60 | 1265.31 | 28887.13 | 4.74 |
| **DACSE-3** | 3458.29 | 5388.22 | 1171.52 | 717.53 | 34198.91 | 2569.54 | 28896.53 | 13.38 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 368.10 | 0.00 | 399.00 | 0.00 | 27353.80 | 0.00 |
| **DACSE-1** | 4608.12 | 6461.55 | 1041.40 | 11.56 | 3212.58 | 2786.90 | 28086.50 | 0.00 |
| **DACSE-2** | 4772.00 | 6820.36 | 1053.52 | 32.77 | 3656.93 | 2416.88 | 28086.50 | 0.00 |
| **DACSE-3** | 3591.74 | 5689.72 | 1029.83 | 17.03 | 2704.51 | 2456.18 | 28086.50 | 0.00 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 447.40 | 0.00 | 37420.80 | 0.00 | 107326.00 | 0.00 |
| **DACSE-1** | 3820.85 | 5685.55 | 8121.50 | 4978.00 | 9898.58 | 6924.85 | 82783.00 | 0.00 |
| **DACSE-2** | 4317.43 | 6499.68 | 5115.29 | 3243.52 | 3965.34 | 2515.05 | 82783.00 | 0.00 |
| **DACSE-3** | 2649.00 | 4584.86 | 7342.20 | 5623.77 | 5570.57 | 3496.16 | 82783.00 | 0.00 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 395.00 | 0.00 | 431.00 | 0.00 | 81322.00 | 0.00 |
| **DACSE-1** | 3655.47 | 5684.26 | 1250.47 | 1272.36 | 26138.60 | 9.14 | 108914.90 | 14005.85 |
| **DACSE-2** | 4154.55 | 6102.09 | 8438.24 | 9742.71 | 26121.48 | 6.19 | 121882.38 | 8481.43 |
| **DACSE-3** | 4310.72 | 6079.61 | 1958.26 | 3989.25 | 26133.77 | 12.87 | 106134.98 | 13324.54 |

Table 96. Dynamic DACSE Solution Cost Results - File 4 (10 Nodes)

| Solution Cost Results - 10 Node DACSE (File 4) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 21410.00 | 0.00 | 21410.00 | 0.00 | 21410.00 | 0.00 |
| **DACSE-1** | 10598.78 | 5653.66 | 9686.54 | 7096.31 | 10452.47 | 6232.16 | 8659.39 | 5678.70 |
| **DACSE-2** | 11180.02 | 5314.27 | 10513.83 | 4864.71 | 11457.00 | 7156.41 | 11639.05 | 4806.48 |
| **DACSE-3** | 10275.90 | 6832.90 | 9391.78 | 8493.88 | 9382.21 | 7529.76 | 9590.34 | 5511.30 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 517.00 | 0.00 | 416.00 | 0.00 | 459.00 | 0.00 |
| **DACSE-1** | 10918.18 | 3628.68 | 13283.81 | 6526.12 | 7154.48 | 5640.49 | 1086.87 | 13.33 |
| **DACSE-2** | 9963.97 | 4770.22 | 14408.51 | 6120.11 | 6056.92 | 5923.33 | 1097.77 | 19.82 |
| **DACSE-3** | 9938.26 | 4927.58 | 10220.40 | 6346.51 | 4251.65 | 4684.05 | 1089.09 | 21.07 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 22416.20 | 0.00 | 25462.00 | 0.00 | 23442.40 | 0.00 |
| **DACSE-1** | 10900.80 | 6225.62 | 16676.24 | 6482.64 | 1175.43 | 2.32 | 29525.62 | 4963.24 |
| **DACSE-2** | 9806.82 | 4418.06 | 17256.75 | 4813.85 | 1172.51 | 5.03 | 29077.45 | 3508.16 |
| **DACSE-3** | 10722.54 | 7079.41 | 17932.85 | 7815.45 | 1175.02 | 3.20 | 28079.03 | 4957.67 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 554.60 | 0.00 | 36434.40 | 0.00 | 91353.20 | 0.00 |
| **DACSE-1** | 11050.67 | 4974.28 | 13882.40 | 3658.83 | 20775.15 | 8615.54 | 81034.30 | 3818.65 |
| **DACSE-2** | 12506.45 | 7758.09 | 10195.77 | 4327.37 | 16538.68 | 8610.95 | 84144.83 | 2508.62 |
| **DACSE-3** | 9619.90 | 5422.58 | 14432.75 | 3321.36 | 20245.34 | 7191.15 | 82707.31 | 4345.80 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 14439.50 | 0.00 | 51358.00 | 0.00 | 38390.00 | 0.00 |
| **DACSE-1** | 11346.09 | 5118.51 | 32514.35 | 7054.03 | 37711.67 | 2040.41 | 62127.53 | 0.69 |
| **DACSE-2** | 13284.12 | 7919.70 | 30383.49 | 5395.79 | 36968.27 | 1.41 | 62127.40 | 0.81 |
| **DACSE-3** | 11331.12 | 5393.01 | 33415.37 | 5991.47 | 37494.04 | 1603.38 | 62127.47 | 0.76 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 26512.40 | 0.00 | 53361.00 | 0.00 | 76328.40 | 0.00 |
| **DACSE-1** | 11468.50 | 4857.56 | 1241.93 | 37.34 | 74818.71 | 8045.43 | 88571.74 | 2503.09 |
| **DACSE-2** | 9943.45 | 4776.51 | 1226.50 | 13.92 | 78104.29 | 10218.45 | 87783.60 | 1245.90 |
| **DACSE-3** | 12003.84 | 6082.48 | 1237.49 | 28.80 | 78304.69 | 9683.22 | 89301.81 | 4274.82 |

**Table 97. Dynamic DACSE Solution Cost Results - File 5 (10 Nodes)**

| Solution Cost Results - 10 Node DACSE (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 466.47 | 0.00 | 466.47 | 0.00 | 466.47 | 0.00 | 466.47 | 0.00 |
| **DACSE-1** | 1091.53 | 32.20 | 1064.80 | 11.16 | 1066.63 | 14.71 | 1062.33 | 8.79 |
| **DACSE-2** | 1080.27 | 29.81 | 1076.83 | 25.29 | 1087.03 | 30.12 | 1086.83 | 28.65 |
| **DACSE-3** | 1085.07 | 30.37 | 1069.43 | 13.12 | 1070.80 | 14.06 | 1065.73 | 10.49 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 466.47 | 0.00 | 468.25 | 0.00 | 464.30 | 0.00 | 476.30 | 0.00 |
| **DACSE-1** | 1151.80 | 29.32 | 10990.53 | 1089.00 | 1194.97 | 20.37 | 1219.37 | 34.21 |
| **DACSE-2** | 1151.87 | 33.64 | 13591.22 | 3727.34 | 1230.67 | 66.65 | 1167.57 | 69.97 |
| **DACSE-3** | 1154.27 | 32.83 | 9625.43 | 2640.22 | 1189.63 | 15.71 | 1212.76 | 48.69 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 466.47 | 0.00 | 520.25 | 0.00 | 55393.80 | 0.00 | 71366.00 | 0.00 |
| **DACSE-1** | 1126.37 | 16.45 | 6307.72 | 733.38 | 22217.30 | 10143.80 | 26893.34 | 4533.17 |
| **DACSE-2** | 1125.37 | 17.08 | 3846.63 | 2537.75 | 22538.59 | 7062.88 | 29376.26 | 8584.59 |
| **DACSE-3** | 1121.47 | 15.67 | 6177.43 | 10.65 | 22882.19 | 10121.23 | 32686.82 | 11167.59 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 466.47 | 0.00 | 474.60 | 0.00 | 15418.70 | 0.00 | 535.00 | 0.00 |
| **DACSE-1** | 1117.07 | 23.29 | 9227.29 | 1677.90 | 18190.00 | 0.00 | 9395.00 | 0.00 |
| **DACSE-2** | 1116.50 | 22.78 | 12326.77 | 2999.51 | 18190.00 | 0.00 | 9395.00 | 0.00 |
| **DACSE-3** | 1115.07 | 24.63 | 10680.70 | 2845.81 | 18190.00 | 0.00 | 9395.00 | 0.00 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 466.47 | 0.00 | 33424.00 | 0.00 | 32391.00 | 0.00 | 58353.00 | 0.00 |
| **DACSE-1** | 1101.10 | 38.50 | 1225.67 | 10.47 | 33071.20 | 13.05 | 86683.00 | 0.00 |
| **DACSE-2** | 1096.23 | 36.61 | 1229.80 | 15.38 | 33758.45 | 1281.82 | 86683.00 | 0.00 |
| **DACSE-3** | 1101.77 | 38.57 | 1222.93 | 11.81 | 33062.73 | 13.88 | 86683.00 | 0.00 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 466.47 | 0.00 | 28388.00 | 0.00 | 10434.00 | 0.00 | 77318.70 | 0.00 |
| **DACSE-1** | 1066.77 | 24.53 | 29139.65 | 4.13 | 59821.80 | 4.38 | 104614.00 | 0.00 |
| **DACSE-2** | 1093.57 | 37.71 | 29156.29 | 20.37 | 59834.57 | 13.55 | 104614.00 | 0.00 |
| DACSE-3 | 1084.07 | 35.37 | 29139.77 | 4.07 | 59821.83 | 4.38 | 104614.00 | 0.00 |

Table 98. Dynamic DACSE Solution Cost Results - File 6 (10 Nodes)

| Solution Cost Results - 10 Node DACSE (File 6) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 367.00 | 0.00 | 367.00 | 0.00 | 367.00 | 0.00 |
| **DACSE-1** | 6246.19 | 5295.33 | 5393.49 | 3822.59 | 3785.58 | 3359.60 | 2887.92 | 3038.21 |
| **DACSE-2** | 3538.97 | 3155.31 | 5744.37 | 4778.88 | 5142.70 | 3779.60 | 5302.70 | 3523.05 |
| **DACSE-3** | 4385.33 | 2999.75 | 3910.26 | 3960.68 | 3535.79 | 3093.91 | 2969.83 | 2719.54 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 352.00 | 0.00 | 368.00 | 0.00 | 356.50 | 0.00 |
| **DACSE-1** | 3890.69 | 3461.28 | 1000.77 | 30.34 | 3367.14 | 2530.80 | 16099.24 | 7251.96 |
| **DACSE-2** | 4740.00 | 4189.98 | 1524.99 | 1521.72 | 2553.94 | 2662.58 | 14506.71 | 7986.87 |
| **DACSE-3** | 4457.48 | 3472.57 | 1175.63 | 917.99 | 2860.93 | 2781.55 | 11324.78 | 6670.30 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 391.50 | 0.00 | 427.75 | 0.00 | 21424.00 | 0.00 |
| **DACSE-1** | 4800.80 | 4095.76 | 1154.07 | 4.63 | 1038.67 | 5.98 | 47936.23 | 17.32 |
| **DACSE-2** | 3462.69 | 2908.64 | 5563.82 | 5292.47 | 1054.27 | 14.49 | 51277.51 | 2413.95 |
| **DACSE-3** | 5591.31 | 5019.82 | 1322.00 | 908.47 | 1033.73 | 14.86 | 47931.51 | 17.43 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 369.50 | 0.00 | 443.40 | 0.00 | 6483.60 | 0.00 |
| **DACSE-1** | 5285.32 | 4260.58 | 8199.03 | 4111.71 | 4143.74 | 2588.35 | 14271.08 | 6772.12 |
| **DACSE-2** | 6317.91 | 3909.78 | 10474.41 | 2784.04 | 5171.19 | 2209.41 | 10181.28 | 8034.51 |
| **DACSE-3** | 3985.66 | 3304.21 | 8553.04 | 3465.07 | 3627.01 | 2604.34 | 13896.62 | 7247.23 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 402.60 | 0.00 | 442.60 | 0.00 | 121214.00 | 0.00 |
| **DACSE-1** | 5579.23 | 3539.04 | 1100.52 | 47.62 | 14061.37 | 1831.27 | 135566.23 | 7.41 |
| **DACSE-2** | 4797.32 | 3757.64 | 1152.67 | 70.83 | 16274.30 | 1445.24 | 135602.90 | 21.23 |
| **DACSE-3** | 4507.52 | 3642.44 | 1092.38 | 37.24 | 15280.57 | 3558.83 | 135566.97 | 7.99 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 396.20 | 0.00 | 21415.00 | 0.00 | 45384.00 | 0.00 |
| **DACSE-1** | 4469.96 | 3136.75 | 1388.57 | 15.29 | 24874.35 | 8255.35 | 51052.00 | 0.00 |
| **DACSE-2** | 5469.09 | 4311.55 | 1399.40 | 11.33 | 21208.13 | 3973.22 | 51114.40 | 24.89 |
| DACSE-3 | 5515.16 | 5079.60 | 1384.05 | 20.96 | 22767.61 | 10172.49 | 51052.00 | 0.00 |

**Table 99. Dynamic DACSE Solution Cost Results - File 7 (10 Nodes)**

| Solution Cost Results - 10 Node DACSE (File 7) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 369.00 | 0.00 | 369.00 | 0.00 | 369.00 | 0.00 |
| **DACSE-1** | 6345.58 | 5619.42 | 7136.33 | 5565.43 | 6330.58 | 5727.74 | 6762.30 | 5054.28 |
| **DACSE-2** | 6275.42 | 5289.46 | 4287.44 | 4698.20 | 5717.78 | 4717.58 | 4691.24 | 4509.80 |
| **DACSE-3** | 3392.61 | 4570.67 | 4803.64 | 4897.21 | 4697.41 | 5649.55 | 4562.07 | 6279.05 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 350.00 | 0.00 | 393.50 | 0.00 | 371.00 | 0.00 |
| **DACSE-1** | 6419.69 | 4997.46 | 1526.55 | 1385.90 | 5571.32 | 1294.71 | 10268.73 | 2528.18 |
| **DACSE-2** | 4991.72 | 4667.33 | 1402.90 | 1217.78 | 5639.58 | 1261.99 | 9950.27 | 3040.06 |
| **DACSE-3** | 6856.98 | 4755.64 | 1921.55 | 1719.06 | 6095.53 | 1981.52 | 7997.40 | 5043.65 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 27319.00 | 0.00 | 25377.00 | 0.00 | 20415.40 | 0.00 |
| **DACSE-1** | 7840.08 | 4762.51 | 27850.10 | 20.87 | 26006.56 | 25.48 | 20936.01 | 13.38 |
| **DACSE-2** | 6244.56 | 5538.69 | 27888.17 | 66.64 | 26031.42 | 29.94 | 20948.75 | 10.35 |
| **DACSE-3** | 4249.47 | 5345.80 | 27833.37 | 22.01 | 25997.87 | 17.23 | 20939.97 | 14.66 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 366.50 | 0.00 | 452.00 | 0.00 | 410.00 | 0.00 |
| **DACSE-1** | 6922.84 | 4365.46 | 1074.84 | 14.43 | 4219.34 | 2097.11 | 6234.15 | 0.00 |
| **DACSE-2** | 7192.45 | 4629.79 | 1073.84 | 16.51 | 7177.06 | 5990.10 | 6068.18 | 909.06 |
| **DACSE-3** | 6659.43 | 6623.40 | 1081.88 | 18.78 | 4927.34 | 1816.15 | 6234.15 | 0.00 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 395.40 | 0.00 | 492.00 | 0.00 | 486.00 | 0.00 |
| **DACSE-1** | 5145.48 | 4978.19 | 1069.20 | 25.56 | 12855.13 | 5543.71 | 42681.12 | 2365.49 |
| **DACSE-2** | 6957.86 | 4811.65 | 1057.36 | 21.34 | 9197.91 | 3353.70 | 43377.08 | 1446.44 |
| **DACSE-3** | 3782.96 | 4706.92 | 1041.28 | 28.40 | 8095.05 | 12.12 | 40953.62 | 4911.07 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 24362.80 | 0.00 | 26458.00 | 0.00 | 104270.00 | 0.00 |
| **DACSE-1** | 6634.37 | 5500.45 | 24964.96 | 6625.02 | 20021.70 | 5.14 | 156467.00 | 0.00 |
| **DACSE-2** | 5706.37 | 4883.51 | 19630.59 | 9741.80 | 20025.45 | 3.80 | 156467.00 | 0.00 |
| DACSE-3 | 5009.38 | 4710.52 | 23818.49 | 9220.45 | 20024.40 | 4.69 | 156467.00 | 0.00 |

**Table 100. Dynamic DACSE Solution Cost Results - File 8 (10 Nodes)**

| Solution Cost Results - 10 Node DACSE (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 11409.00 | 0.00 | 11409.00 | 0.00 | 11409.00 | 0.00 |
| **DACSE-1** | 2030.81 | 2500.47 | 1613.15 | 1499.65 | 1754.45 | 1609.49 | 2145.97 | 1838.95 |
| **DACSE-2** | 2361.39 | 1894.93 | 2565.12 | 2299.16 | 2705.54 | 2681.14 | 2208.59 | 1850.58 |
| **DACSE-3** | 1553.18 | 1365.84 | 2289.49 | 1902.83 | 2026.18 | 1776.24 | 2019.60 | 1780.24 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 11371.00 | 0.00 | 37331.40 | 0.00 | 35334.00 | 0.00 |
| **DACSE-1** | 2505.56 | 2331.52 | 5357.28 | 5490.43 | 37955.57 | 50.28 | 35851.53 | 17.02 |
| **DACSE-2** | 1484.75 | 1199.62 | 5596.18 | 6054.90 | 37951.70 | 36.98 | 35893.62 | 51.37 |
| **DACSE-3** | 2024.73 | 1712.55 | 5798.41 | 7624.35 | 37935.30 | 28.83 | 35843.60 | 14.31 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 426.00 | 0.00 | 4427.00 | 0.00 | 36367.00 | 0.00 |
| **DACSE-1** | 2397.60 | 2297.52 | 2898.14 | 6609.58 | 12101.13 | 14.01 | 28038.21 | 12.88 |
| **DACSE-2** | 1860.81 | 2068.32 | 2364.55 | 2564.44 | 12100.60 | 10.38 | 28530.03 | 2726.50 |
| **DACSE-3** | 1727.67 | 1501.78 | 1165.81 | 1084.06 | 12102.50 | 14.14 | 28038.33 | 19.19 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 420.80 | 0.00 | 13401.00 | 0.00 | 5445.00 | 0.00 |
| **DACSE-1** | 2004.89 | 1677.10 | 1851.88 | 3055.11 | 29338.86 | 4832.54 | 64844.80 | 727.74 |
| **DACSE-2** | 2123.30 | 2181.99 | 2787.03 | 2279.23 | 36318.21 | 8568.47 | 62849.90 | 2021.84 |
| **DACSE-3** | 2369.40 | 2192.13 | 2115.82 | 3134.56 | 28799.82 | 6204.39 | 64316.60 | 1510.90 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 8418.00 | 0.00 | 454.00 | 0.00 | 30441.00 | 0.00 |
| **DACSE-1** | 2053.65 | 1778.45 | 1122.44 | 58.31 | 1293.08 | 915.59 | 3819.20 | 1134.87 |
| **DACSE-2** | 1432.47 | 1197.78 | 1121.36 | 68.67 | 3998.53 | 2561.83 | 3220.40 | 1467.46 |
| **DACSE-3** | 1910.33 | 2045.48 | 1083.32 | 55.24 | 1292.17 | 921.34 | 3420.82 | 1394.22 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 463.00 | 0.00 | 76254.00 | 0.00 | 108289.00 | 0.00 |
| **DACSE-1** | 1892.43 | 1624.07 | 15716.84 | 5403.93 | 39882.37 | 2514.66 | 103574.40 | 1660.71 |
| **DACSE-2** | 2529.19 | 1953.19 | 16423.39 | 6598.43 | 37887.18 | 1890.99 | 102758.00 | 0.00 |
| DACSE-3 | 2159.54 | 1779.43 | 13773.23 | 4877.36 | 38718.17 | 2392.21 | 102758.00 | 0.00 |

**Table 101. Dynamic DACSE Solution Cost Results - File 9 (10 Nodes)**

| Solution Cost Results - 10 Node DACSE (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 410.00 | 0.00 | 410.00 | 0.00 | 410.00 | 0.00 |
| **DACSE-1** | 1065.13 | 17.09 | 1063.67 | 23.79 | 1060.47 | 25.20 | 1203.73 | 735.16 |
| **DACSE-2** | 1063.83 | 21.25 | 1067.60 | 18.85 | 1063.63 | 22.88 | 1062.47 | 26.71 |
| **DACSE-3** | 1058.43 | 22.27 | 1059.86 | 36.27 | 1062.68 | 41.50 | 1365.03 | 1143.80 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 384.40 | 0.00 | 355.83 | 0.00 | 321.00 | 0.00 |
| **DACSE-1** | 1020.70 | 25.77 | 989.37 | 17.94 | 1052.37 | 25.68 | 1021.13 | 19.25 |
| **DACSE-2** | 1144.90 | 735.44 | 987.30 | 9.80 | 1070.10 | 27.19 | 1040.30 | 22.01 |
| **DACSE-3** | 1149.75 | 734.52 | 983.83 | 14.13 | 1050.70 | 30.28 | 1021.30 | 23.76 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 430.00 | 0.00 | 24529.80 | 0.00 | 22454.80 | 0.00 |
| **DACSE-1** | 1102.57 | 31.85 | 11028.02 | 6357.99 | 27109.50 | 29.07 | 26121.71 | 1.17 |
| **DACSE-2** | 1102.60 | 13.47 | 6766.89 | 7719.73 | 27480.76 | 2001.92 | 26116.07 | 5.56 |
| **DACSE-3** | 1257.00 | 897.01 | 12159.52 | 7345.17 | 27242.41 | 729.57 | 26219.18 | 542.83 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 392.75 | 0.00 | 432.00 | 0.00 | 531.80 | 0.00 |
| **DACSE-1** | 1115.60 | 13.33 | 1127.65 | 22.05 | 2750.92 | 3168.85 | 6199.84 | 3481.02 |
| **DACSE-2** | 1118.00 | 15.36 | 1131.13 | 20.82 | 3248.07 | 3570.52 | 13428.78 | 5936.96 |
| **DACSE-3** | 1118.60 | 19.41 | 1285.12 | 904.21 | 2548.39 | 3010.94 | 5835.12 | 5026.83 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 426.00 | 0.00 | 547.25 | 0.00 | 89378.00 | 0.00 |
| **DACSE-1** | 1101.17 | 30.11 | 4450.70 | 3849.38 | 1376.63 | 571.77 | 58063.00 | 7.61 |
| **DACSE-2** | 1098.67 | 31.18 | 6042.60 | 2938.35 | 1322.71 | 67.37 | 58095.43 | 18.90 |
| **DACSE-3** | 1092.73 | 30.26 | 4487.38 | 4367.34 | 1277.94 | 64.06 | 58062.00 | 9.15 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 437.00 | 0.00 | 21401.00 | 0.00 | 92267.00 | 0.00 |
| **DACSE-1** | 1074.33 | 16.58 | 10487.96 | 5030.98 | 27089.00 | 12.65 | 125711.00 | 0.00 |
| **DACSE-2** | 1077.50 | 13.29 | 9637.98 | 966.38 | 27129.30 | 8.35 | 125711.00 | 0.00 |
| DACSE-3 | 1060.78 | 26.21 | 10745.17 | 6977.78 | 27095.40 | 17.04 | 125711.00 | 0.00 |

**Table 102. Dynamic DACSE Solution Cost Results - File 10 (10 Nodes)**

| Solution Cost Results - 10 Node DACSE (File 10) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 30328.40 | 0.00 | 30328.40 | 0.00 | 30328.40 | 0.00 |
| **DACSE-1** | 30871.67 | 39.37 | 30863.07 | 32.31 | 31129.20 | 1012.22 | 30850.30 | 37.24 |
| **DACSE-2** | 30871.80 | 28.56 | 30872.00 | 34.75 | 31002.43 | 727.55 | 31004.93 | 727.37 |
| **DACSE-3** | 30886.83 | 29.89 | 31006.37 | 727.17 | 30877.20 | 38.00 | 31136.63 | 1010.74 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 29334.00 | 0.00 | 26342.60 | 0.00 | 26360.30 | 0.00 |
| **DACSE-1** | 31042.70 | 724.54 | 32867.24 | 2486.72 | 28980.68 | 2254.39 | 39287.97 | 996.58 |
| **DACSE-2** | 30911.60 | 35.93 | 30887.70 | 2022.96 | 28738.28 | 2325.76 | 40177.67 | 1561.42 |
| **DACSE-3** | 30902.43 | 41.80 | 31708.00 | 2444.00 | 28654.10 | 2159.51 | 39514.49 | 1814.99 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 377.00 | 0.00 | 479.40 | 0.00 | 52338.00 | 0.00 |
| **DACSE-1** | 30845.73 | 29.40 | 920.79 | 15.71 | 10057.67 | 8600.54 | 22058.77 | 6.98 |
| **DACSE-2** | 30843.33 | 37.98 | 924.48 | 17.59 | 11935.23 | 9507.66 | 22071.97 | 13.27 |
| **DACSE-3** | 30839.93 | 41.24 | 911.64 | 6.84 | 5969.73 | 7724.42 | 22054.93 | 9.45 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 382.40 | 0.00 | 388.00 | 0.00 | 30351.20 | 0.00 |
| **DACSE-1** | 30942.60 | 46.91 | 22873.47 | 4565.54 | 9502.95 | 7005.21 | 1117.20 | 2.31 |
| **DACSE-2** | 30939.90 | 38.00 | 24075.50 | 2.33 | 15088.38 | 5934.70 | 1138.80 | 24.16 |
| **DACSE-3** | 30970.77 | 39.86 | 22701.65 | 4608.83 | 8806.64 | 6207.40 | 1121.37 | 10.70 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 35389.00 | 0.00 | 450.00 | 0.00 | 22495.00 | 0.00 |
| **DACSE-1** | 31065.30 | 717.23 | 35994.93 | 22.96 | 1171.83 | 14.28 | 43255.33 | 18.22 |
| **DACSE-2** | 30924.30 | 42.21 | 35969.38 | 19.89 | 1164.07 | 8.59 | 43265.47 | 9.64 |
| **DACSE-3** | 30917.61 | 38.85 | 35982.15 | 23.50 | 1170.73 | 15.56 | 43250.27 | 19.28 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 419.67 | 0.00 | 29429.00 | 0.00 | 63365.60 | 0.00 |
| **DACSE-1** | 30974.23 | 35.77 | 3138.12 | 3106.48 | 30616.45 | 1280.87 | 106828.00 | 0.00 |
| **DACSE-2** | 30970.00 | 45.91 | 3316.24 | 3605.49 | 30361.23 | 48.79 | 106828.00 | 0.00 |
| DACSE-3 | 30969.17 | 33.78 | 3666.61 | 3007.84 | 30631.40 | 1279.29 | 106828.00 | 0.00 |

Table 103. Dynamic DACSS Solution Cost Results - File 1 (10 Nodes)

| Solution Cost Results - 10 Node DACSS (File 1) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 |
| **DACSS-1** | 1099.97 | 39.15 | 1103.36 | 36.91 | 1102.98 | 41.74 | 1098.75 | 38.76 |
| **DACSS-2** | 1115.37 | 26.76 | 1111.88 | 36.04 | 1107.99 | 31.52 | 1115.85 | 32.48 |
| **DACSS-3** | 1038.57 | 50.15 | 1036.02 | 36.08 | 1045.10 | 41.05 | 1034.51 | 36.50 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 471.25 | 0.00 | 5414.67 | 0.00 | 402.00 | 0.00 |
| **DACSS-1** | 1115.69 | 30.58 | 1043.25 | 26.74 | 1080.44 | 32.28 | 992.34 | 17.33 |
| **DACSS-2** | 1115.75 | 38.20 | 1049.68 | 27.33 | 1072.19 | 29.73 | 990.98 | 23.15 |
| **DACSS-3** | 1050.74 | 43.33 | 1014.40 | 27.07 | 1065.36 | 29.94 | 967.01 | 26.32 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| **MILP** | 519.00 | 0.00 | 612.00 | 0.00 | 52380.00 | 0.00 | 56411.00 | 0.00 |
| **DACSS-1** | 1132.98 | 46.23 | 1057.80 | 29.54 | 7666.78 | 4409.61 | 24944.50 | 10.16 |
| **DACSS-2** | 1162.98 | 27.60 | 1063.92 | 23.17 | 9995.61 | 4192.59 | 24943.84 | 9.07 |
| **DACSS-3** | 1080.48 | 71.58 | 1052.47 | 14.57 | 5320.21 | 3178.12 | 24931.89 | 12.04 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 56331.70 | 0.00 | 31446.80 | 0.00 | 49381.00 | 0.00 |
| **DACSS-1** | 1154.54 | 36.52 | 31022.80 | 566.44 | 41259.62 | 5836.14 | 2238.44 | 3097.31 |
| **DACSS-2** | 1177.70 | 42.57 | 30951.55 | 46.86 | 41155.09 | 6208.74 | 3169.77 | 4050.40 |
| **DACSS-3** | 1129.55 | 65.85 | 30899.86 | 18.77 | 34281.31 | 3718.68 | 1203.19 | 52.08 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 487.00 | 0.00 | 62357.00 | 0.00 | 74354.20 | 0.00 |
| **DACSS-1** | 1096.33 | 55.24 | 1152.57 | 13.55 | 28982.05 | 23.62 | 74366.84 | 2333.97 |
| **DACSS-2** | 1120.49 | 24.21 | 1153.41 | 14.59 | 28996.31 | 18.69 | 74588.11 | 2269.60 |
| **DACSS-3** | 1034.21 | 41.27 | 1137.56 | 20.07 | 28952.71 | 20.58 | 72250.28 | 856.31 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 14439.00 | 0.00 | 20375.00 | 0.00 | 27360.70 | 0.00 |
| **DACSS-1** | 1116.09 | 37.34 | 1133.96 | 29.28 | 20918.75 | 25.01 | 19026.16 | 7.72 |
| **DACSS-2** | 1118.62 | 37.38 | 1130.79 | 31.92 | 20922.09 | 28.02 | 19027.46 | 7.09 |
| **DACSS-3** | 1065.56 | 46.77 | 1102.88 | 31.44 | 20883.29 | 21.67 | 19017.34 | 12.60 |

**Table 104. Dynamic DACSS Solution Cost Results - File 2 (10 Nodes)**

| Solution Cost Results - 10 Node DACSS (File 2) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 | 519.00 | 0.00 |
| **DACSS-1** | 944.26 | 39.76 | 950.24 | 13.28 | 944.93 | 16.27 | 945.42 | 12.28 |
| **DACSS-2** | 947.02 | 19.40 | 949.05 | 16.52 | 950.02 | 14.91 | 945.36 | 17.63 |
| **DACSS-3** | 928.12 | 18.58 | 925.91 | 19.41 | 925.63 | 11.53 | 929.18 | 15.32 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 471.25 | 0.00 | 5414.67 | 0.00 | 402.00 | 0.00 |
| **DACSS-1** | 951.88 | 13.59 | 952.94 | 12.86 | 984.06 | 13.49 | 921.35 | 17.75 |
| **DACSS-2** | 947.35 | 18.20 | 950.64 | 14.15 | 969.47 | 19.37 | 924.95 | 11.67 |
| **DACSS-3** | 933.63 | 13.44 | 943.92 | 14.26 | 945.68 | 20.34 | 900.24 | 17.27 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| **MILP** | 519.00 | 0.00 | 612.00 | 0.00 | 52380.00 | 0.00 | 56411.00 | 0.00 |
| **DACSS-1** | 956.38 | 18.85 | 942.47 | 14.61 | 1050.19 | 15.47 | 24983.73 | 44.78 |
| **DACSS-2** | 960.34 | 22.48 | 947.05 | 13.46 | 1048.60 | 21.05 | 25030.73 | 27.08 |
| **DACSS-3** | 942.21 | 36.26 | 930.05 | 19.35 | 1036.44 | 21.40 | 24977.28 | 27.33 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 56331.70 | 0.00 | 31446.80 | 0.00 | 49381.00 | 0.00 |
| **DACSS-1** | 953.46 | 11.87 | 1113.98 | 31.71 | 1070.85 | 28.94 | 31696.94 | 1864.98 |
| **DACSS-2** | 951.11 | 14.61 | 1111.80 | 22.54 | 1060.08 | 35.10 | 31795.01 | 2430.11 |
| **DACSS-3** | 937.33 | 20.03 | 1075.09 | 36.22 | 1059.84 | 27.66 | 29643.66 | 2054.45 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 487.00 | 0.00 | 62357.00 | 0.00 | 74354.20 | 0.00 |
| **DACSS-1** | 971.35 | 15.81 | 13349.40 | 1166.87 | 14788.88 | 2484.49 | 62837.09 | 3336.97 |
| **DACSS-2** | 971.18 | 12.52 | 13750.94 | 892.36 | 14370.05 | 2958.04 | 65780.53 | 5742.76 |
| **DACSS-3** | 953.01 | 31.91 | 11966.83 | 1757.90 | 12389.91 | 2827.14 | 60166.34 | 3949.49 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 519.00 | 0.00 | 14439.00 | 0.00 | 20375.00 | 0.00 | 27360.70 | 0.00 |
| **DACSS-1** | 904.72 | 15.44 | 1047.04 | 17.32 | 1142.85 | 25.36 | 39365.47 | 3452.42 |
| **DACSS-2** | 903.91 | 14.79 | 1050.54 | 22.08 | 1136.89 | 28.65 | 40565.80 | 4714.88 |
| **DACSS-3** | 892.60 | 19.67 | 1026.73 | 17.50 | 1129.77 | 34.05 | 38699.30 | 2540.82 |

Table 105. Dynamic DACSS Solution Cost Results - File 3 (10 Nodes)

| Solution Cost Results - 10 Node DACSS (File 3) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 372.40 | 0.00 | 372.40 | 0.00 | 372.40 | 0.00 |
| **DACSS-1** | 900.01 | 14.83 | 900.30 | 10.40 | 900.57 | 11.14 | 905.14 | 10.21 |
| **DACSS-2** | 903.03 | 17.44 | 900.63 | 18.36 | 902.11 | 12.07 | 903.27 | 16.65 |
| **DACSS-3** | 886.43 | 12.32 | 884.24 | 15.43 | 882.56 | 16.57 | 886.35 | 14.26 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 339.60 | 0.00 | 350.67 | 0.00 | 345.50 | 0.00 |
| **DACSS-1** | 916.86 | 20.12 | 932.49 | 18.85 | 948.87 | 14.62 | 961.27 | 11.43 |
| **DACSS-2** | 922.97 | 13.33 | 936.21 | 17.33 | 949.37 | 14.51 | 954.63 | 20.00 |
| **DACSS-3** | 896.87 | 28.76 | 905.64 | 21.05 | 931.78 | 19.07 | 931.33 | 17.09 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| **MILP** | 372.40 | 0.00 | 383.60 | 0.00 | 30398.00 | 0.00 | 28456.80 | 0.00 |
| **DACSS-1** | 955.07 | 15.11 | 919.41 | 11.55 | 30885.93 | 11.23 | 28861.10 | 10.74 |
| **DACSS-2** | 957.61 | 12.00 | 913.60 | 12.13 | 30892.45 | 11.75 | 28857.57 | 13.42 |
| **DACSS-3** | 941.05 | 27.09 | 906.91 | 17.90 | 30870.16 | 12.08 | 28849.20 | 9.58 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 368.10 | 0.00 | 399.00 | 0.00 | 27353.80 | 0.00 |
| **DACSS-1** | 941.46 | 11.78 | 951.77 | 8.65 | 1008.03 | 14.03 | 27972.08 | 16.80 |
| **DACSS-2** | 938.02 | 21.88 | 955.10 | 8.98 | 1001.17 | 17.39 | 28025.50 | 17.63 |
| **DACSS-3** | 929.80 | 15.88 | 936.83 | 15.01 | 983.72 | 21.25 | 27969.59 | 29.06 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 447.40 | 0.00 | 37420.80 | 0.00 | 107326.00 | 0.00 |
| **DACSS-1** | 992.33 | 20.55 | 1000.21 | 13.14 | 1182.39 | 30.66 | 66230.92 | 1491.23 |
| **DACSS-2** | 1000.44 | 11.68 | 997.18 | 19.46 | 1178.18 | 31.78 | 67508.07 | 902.59 |
| **DACSS-3** | 977.47 | 21.34 | 989.17 | 18.98 | 1160.76 | 26.73 | 64946.09 | 539.00 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 372.40 | 0.00 | 395.00 | 0.00 | 431.00 | 0.00 | 81322.00 | 0.00 |
| **DACSS-1** | 985.95 | 14.26 | 951.79 | 8.78 | 21063.44 | 20.46 | 90414.45 | 1202.32 |
| **DACSS-2** | 985.62 | 15.19 | 951.27 | 11.30 | 21076.91 | 22.26 | 90027.06 | 2584.32 |
| **DACSS-3** | 964.23 | 23.48 | 946.91 | 10.97 | 21028.70 | 28.63 | 88046.50 | 1834.20 |

**Table 106. Dynamic DACSS Solution Cost Results - File 4 (10 Nodes)**

| Solution Cost Results - 10 Node DACSS (File 4) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 21410.00 | 0.00 | 21410.00 | 0.00 | 21410.00 | 0.00 |
| **DACSS-1** | 978.30 | 17.46 | 979.71 | 18.96 | 980.59 | 19.44 | 986.31 | 12.31 |
| **DACSS-2** | 978.70 | 14.97 | 980.73 | 18.95 | 982.49 | 17.53 | 980.93 | 13.50 |
| **DACSS-3** | 975.01 | 18.27 | 973.69 | 19.99 | 970.87 | 16.36 | 973.11 | 16.81 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 517.00 | 0.00 | 416.00 | 0.00 | 459.00 | 0.00 |
| **DACSS-1** | 1049.80 | 19.00 | 1022.64 | 19.07 | 986.93 | 13.68 | 988.01 | 11.62 |
| **DACSS-2** | 1052.76 | 15.96 | 1023.52 | 21.81 | 983.03 | 11.31 | 992.43 | 12.27 |
| **DACSS-3** | 1031.36 | 26.06 | 1012.33 | 17.75 | 973.18 | 15.64 | 977.04 | 16.20 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| **MILP** | 21410.00 | 0.00 | 22416.20 | 0.00 | 25462.00 | 0.00 | 23442.40 | 0.00 |
| **DACSS-1** | 997.80 | 19.30 | 1069.92 | 21.96 | 1085.09 | 21.76 | 23923.80 | 42.43 |
| **DACSS-2** | 1002.51 | 17.03 | 1070.45 | 24.41 | 1075.37 | 15.06 | 23938.84 | 37.57 |
| **DACSS-3** | 990.19 | 28.58 | 1066.73 | 21.91 | 1063.40 | 23.42 | 23916.11 | 37.20 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 554.60 | 0.00 | 36434.40 | 0.00 | 91353.20 | 0.00 |
| **DACSS-1** | 1023.24 | 19.74 | 1079.80 | 38.10 | 11259.03 | 913.53 | 70922.68 | 4138.99 |
| **DACSS-2** | 1025.51 | 22.85 | 1076.51 | 35.35 | 11327.94 | 754.14 | 71315.63 | 3945.34 |
| **DACSS-3** | 1009.31 | 19.53 | 1057.89 | 32.09 | 8964.88 | 3631.23 | 67035.64 | 4608.59 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 14439.50 | 0.00 | 51358.00 | 0.00 | 38390.00 | 0.00 |
| **DACSS-1** | 1050.94 | 65.31 | 14818.03 | 4072.52 | 33840.39 | 757.13 | 49761.75 | 4317.92 |
| **DACSS-2** | 1063.17 | 12.48 | 16537.35 | 3188.51 | 33529.61 | 1126.44 | 48762.42 | 3558.88 |
| **DACSS-3** | 1059.02 | 16.96 | 10482.21 | 3062.72 | 33615.98 | 1020.88 | 45641.87 | 2664.13 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 21410.00 | 0.00 | 26512.40 | 0.00 | 53361.00 | 0.00 | 76328.40 | 0.00 |
| **DACSS-1** | 1062.64 | 15.70 | 1045.66 | 11.32 | 68810.16 | 7.26 | 81360.03 | 3375.78 |
| **DACSS-2** | 1060.65 | 21.76 | 1049.08 | 9.37 | 68812.57 | 7.50 | 80792.37 | 3251.67 |
| **DACSS-3** | 1054.98 | 17.17 | 1037.71 | 14.49 | 68474.20 | 1262.93 | 78224.43 | 2398.84 |

## Table 107. Dynamic DACSS Solution Cost Results - File 5 (10 Nodes)

| Solution Cost Results - 10 Node DACSS (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 466.47 | 0.00 | 466.47 | 0.00 | 466.47 | 0.00 | 466.47 | 0.00 |
| **DACSS-1** | 975.37 | 18.09 | 982.28 | 13.67 | 978.84 | 18.69 | 978.37 | 21.12 |
| **DACSS-2** | 978.62 | 18.16 | 983.68 | 15.40 | 980.34 | 15.81 | 979.65 | 17.77 |
| **DACSS-3** | 945.58 | 18.51 | 943.17 | 24.15 | 947.90 | 16.76 | 941.17 | 19.37 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 466.47 | 0.00 | 468.25 | 0.00 | 464.30 | 0.00 | 476.30 | 0.00 |
| **DACSS-1** | 1031.14 | 17.31 | 1045.42 | 20.94 | 997.67 | 11.70 | 981.52 | 11.34 |
| **DACSS-2** | 1036.41 | 19.20 | 1040.32 | 18.02 | 1001.00 | 19.44 | 979.59 | 16.32 |
| **DACSS-3** | 975.37 | 62.94 | 1004.65 | 21.38 | 985.27 | 15.15 | 972.78 | 14.07 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| **MILP** | 466.47 | 0.00 | 520.25 | 0.00 | 55393.80 | 0.00 | 71366.00 | 0.00 |
| **DACSS-1** | 1020.84 | 19.45 | 1119.73 | 18.88 | 11967.66 | 881.32 | 22787.28 | 260.71 |
| **DACSS-2** | 1015.14 | 19.62 | 1125.30 | 16.12 | 12103.54 | 536.31 | 22883.20 | 218.01 |
| **DACSS-3** | 1005.39 | 26.44 | 1111.79 | 19.96 | 11209.32 | 1814.68 | 22671.02 | 237.46 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 466.47 | 0.00 | 474.60 | 0.00 | 15418.70 | 0.00 | 535.00 | 0.00 |
| **DACSS-1** | 997.38 | 20.43 | 1061.52 | 23.74 | 1187.89 | 25.48 | 8338.33 | 20.19 |
| **DACSS-2** | 997.37 | 23.50 | 1060.01 | 26.15 | 1200.65 | 32.36 | 8344.30 | 24.19 |
| **DACSS-3** | 972.31 | 23.06 | 1038.21 | 28.40 | 1176.61 | 26.02 | 8318.77 | 17.67 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 466.47 | 0.00 | 33424.00 | 0.00 | 32391.00 | 0.00 | 58353.00 | 0.00 |
| **DACSS-1** | 983.86 | 19.14 | 1066.32 | 19.20 | 32954.57 | 25.89 | 86675.87 | 1.74 |
| **DACSS-2** | 986.78 | 21.93 | 1067.07 | 19.26 | 32976.27 | 20.22 | 86675.07 | 3.27 |
| **DACSS-3** | 960.29 | 38.26 | 1048.47 | 30.61 | 32941.93 | 26.56 | 86671.40 | 6.86 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 466.47 | 0.00 | 28388.00 | 0.00 | 10434.00 | 0.00 | 77318.70 | 0.00 |
| **DACSS-1** | 962.73 | 17.52 | 28959.88 | 35.48 | 59765.53 | 7.20 | 104605.33 | 2.37 |
| **DACSS-2** | 965.42 | 15.25 | 28982.84 | 23.74 | 59767.53 | 6.28 | 104605.87 | 0.73 |
| **DACSS-3** | 953.41 | 20.08 | 28945.06 | 22.66 | 59761.70 | 6.44 | 104600.00 | 5.83 |

Table 108. Dynamic DACSS Solution Cost Results - File 6 (10 Nodes)

| Solution Cost Results - 10 Node DACSS (File 6) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 367.00 | 0.00 | 367.00 | 0.00 | 367.00 | 0.00 |
| **DACSS-1** | 940.43 | 15.10 | 936.88 | 19.26 | 939.22 | 20.01 | 937.62 | 15.70 |
| **DACSS-2** | 943.24 | 17.99 | 944.69 | 18.61 | 939.45 | 17.35 | 943.09 | 16.60 |
| **DACSS-3** | 916.60 | 15.08 | 918.77 | 15.90 | 917.87 | 19.03 | 919.84 | 13.42 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 352.00 | 0.00 | 368.00 | 0.00 | 356.50 | 0.00 |
| **DACSS-1** | 975.66 | 19.20 | 905.83 | 10.96 | 910.69 | 14.13 | 928.57 | 16.35 |
| **DACSS-2** | 976.60 | 16.84 | 907.43 | 11.50 | 912.89 | 11.14 | 939.02 | 14.33 |
| **DACSS-3** | 948.83 | 13.70 | 891.64 | 17.06 | 904.70 | 15.32 | 923.18 | 19.69 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| **MILP** | 367.00 | 0.00 | 391.50 | 0.00 | 427.75 | 0.00 | 21424.00 | 0.00 |
| **DACSS-1** | 947.92 | 15.37 | 1021.01 | 23.22 | 950.93 | 12.29 | 47823.19 | 22.37 |
| **DACSS-2** | 946.56 | 18.48 | 1027.43 | 17.54 | 949.77 | 11.28 | 47843.60 | 9.49 |
| **DACSS-3** | 933.77 | 18.30 | 991.63 | 27.94 | 939.72 | 12.55 | 47812.19 | 17.30 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 369.50 | 0.00 | 443.40 | 0.00 | 6483.60 | 0.00 |
| **DACSS-1** | 997.67 | 19.14 | 984.88 | 17.54 | 1124.74 | 27.61 | 1109.63 | 17.85 |
| **DACSS-2** | 1003.07 | 16.92 | 989.02 | 17.69 | 1124.25 | 25.09 | 1106.71 | 16.71 |
| **DACSS-3** | 965.81 | 16.08 | 964.73 | 16.98 | 1091.64 | 28.73 | 1100.69 | 15.58 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 402.60 | 0.00 | 442.60 | 0.00 | 121214.00 | 0.00 |
| **DACSS-1** | 1021.70 | 15.39 | 992.93 | 13.38 | 1210.68 | 30.87 | 135511.10 | 3.30 |
| **DACSS-2** | 1023.30 | 17.27 | 991.97 | 15.45 | 1197.72 | 38.17 | 135524.33 | 10.26 |
| **DACSS-3** | 990.10 | 30.70 | 976.93 | 17.37 | 1178.05 | 46.85 | 135510.33 | 2.06 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 367.00 | 0.00 | 396.20 | 0.00 | 21415.00 | 0.00 | 45384.00 | 0.00 |
| **DACSS-1** | 1035.60 | 16.82 | 1142.51 | 30.30 | 1138.33 | 1.80 | 50975.20 | 0.00 |
| **DACSS-2** | 1043.37 | 17.24 | 1137.13 | 32.14 | 1138.00 | 1.90 | 50975.20 | 0.00 |
| **DACSS-3** | 1018.46 | 26.41 | 1105.85 | 39.41 | 1140.62 | 17.60 | 50975.20 | 0.00 |

## Table 109. Dynamic DACSS Solution Cost Results - File 7 (10 Nodes)

| Solution Cost Results - 10 Node DACSS (File 7) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 369.00 | 0.00 | 369.00 | 0.00 | 369.00 | 0.00 |
| **DACSS-1** | 869.21 | 17.07 | 861.83 | 22.21 | 864.46 | 17.61 | 867.99 | 17.30 |
| **DACSS-2** | 864.61 | 15.75 | 864.42 | 16.38 | 866.89 | 12.76 | 867.58 | 12.53 |
| **DACSS-3** | 858.81 | 18.73 | 858.23 | 15.75 | 860.44 | 18.40 | 856.18 | 15.56 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 350.00 | 0.00 | 393.50 | 0.00 | 371.00 | 0.00 |
| **DACSS-1** | 919.80 | 21.98 | 913.88 | 12.54 | 885.27 | 18.42 | 861.27 | 12.03 |
| **DACSS-2** | 914.54 | 24.91 | 916.35 | 10.18 | 889.29 | 16.25 | 860.62 | 11.58 |
| **DACSS-3** | 912.69 | 25.75 | 892.93 | 15.48 | 862.21 | 17.84 | 855.09 | 12.60 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| **MILP** | 369.00 | 0.00 | 27319.00 | 0.00 | 25377.00 | 0.00 | 20415.40 | 0.00 |
| **DACSS-1** | 894.47 | 14.54 | 27775.32 | 14.18 | 25968.53 | 15.92 | 20889.38 | 13.02 |
| **DACSS-2** | 896.08 | 13.62 | 27859.73 | 18.89 | 25949.27 | 17.32 | 20892.23 | 14.91 |
| **DACSS-3** | 887.17 | 17.78 | 27775.67 | 13.32 | 25927.07 | 19.80 | 20868.20 | 18.30 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 366.50 | 0.00 | 452.00 | 0.00 | 410.00 | 0.00 |
| **DACSS-1** | 869.04 | 30.17 | 967.31 | 11.95 | 999.36 | 16.94 | 1115.01 | 37.00 |
| **DACSS-2** | 882.12 | 26.89 | 963.03 | 14.98 | 998.53 | 16.86 | 1107.02 | 31.70 |
| **DACSS-3** | 871.55 | 27.62 | 930.00 | 18.70 | 983.23 | 18.37 | 1070.94 | 24.77 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 395.40 | 0.00 | 492.00 | 0.00 | 486.00 | 0.00 |
| **DACSS-1** | 922.14 | 15.31 | 964.31 | 14.77 | 3573.91 | 1545.44 | 25949.83 | 1888.66 |
| **DACSS-2** | 923.04 | 16.50 | 963.40 | 10.62 | 3536.61 | 1629.87 | 27447.09 | 2527.48 |
| **DACSS-3** | 919.45 | 13.18 | 954.42 | 21.28 | 2896.98 | 1809.69 | 25071.77 | 41.94 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 369.00 | 0.00 | 24362.80 | 0.00 | 26458.00 | 0.00 | 104270.00 | 0.00 |
| **DACSS-1** | 952.38 | 17.82 | 1067.64 | 19.26 | 19996.57 | 8.99 | 156463.93 | 0.94 |
| **DACSS-2** | 952.80 | 14.03 | 1072.67 | 15.77 | 19998.40 | 10.07 | 156463.80 | 0.81 |
| **DACSS-3** | 944.72 | 17.24 | 1055.74 | 22.44 | 19988.10 | 11.14 | 156463.43 | 0.86 |

Table 110. Dynamic DACSS Solution Cost Results - File 8 (10 Nodes)

| Solution Cost Results - 10 Node DACSS (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 11409.00 | 0.00 | 11409.00 | 0.00 | 11409.00 | 0.00 |
| **DACSS-1** | 871.04 | 11.79 | 866.68 | 9.69 | 868.63 | 14.27 | 868.47 | 11.98 |
| **DACSS-2** | 866.40 | 14.26 | 870.99 | 8.75 | 869.50 | 7.86 | 868.62 | 14.48 |
| **DACSS-3** | 854.93 | 12.29 | 853.90 | 13.02 | 853.07 | 11.76 | 852.53 | 13.82 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 11371.00 | 0.00 | 37331.40 | 0.00 | 35334.00 | 0.00 |
| **DACSS-1** | 943.94 | 13.42 | 911.19 | 14.17 | 37816.58 | 12.96 | 35782.27 | 9.01 |
| **DACSS-2** | 944.46 | 14.37 | 916.29 | 19.27 | 37827.90 | 9.80 | 35779.64 | 8.20 |
| **DACSS-3** | 933.47 | 13.76 | 908.97 | 17.43 | 37803.64 | 20.82 | 35773.34 | 9.67 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| **MILP** | 11409.00 | 0.00 | 426.00 | 0.00 | 4427.00 | 0.00 | 36367.00 | 0.00 |
| **DACSS-1** | 879.44 | 9.80 | 884.92 | 37.23 | 1182.69 | 553.85 | 27882.12 | 20.69 |
| **DACSS-2** | 878.78 | 15.37 | 888.31 | 30.55 | 1095.48 | 24.08 | 27942.14 | 33.35 |
| **DACSS-3** | 873.06 | 10.07 | 882.41 | 33.17 | 1052.22 | 46.72 | 27881.70 | 35.00 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 420.80 | 0.00 | 13401.00 | 0.00 | 5445.00 | 0.00 |
| **DACSS-1** | 919.09 | 18.39 | 954.82 | 12.14 | 1116.57 | 26.27 | 35866.16 | 1452.07 |
| **DACSS-2** | 911.21 | 10.64 | 955.78 | 14.45 | 1118.72 | 22.52 | 39842.95 | 4114.93 |
| **DACSS-3** | 903.49 | 18.36 | 935.52 | 17.98 | 1083.83 | 49.41 | 35656.04 | 1925.95 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 8418.00 | 0.00 | 454.00 | 0.00 | 30441.00 | 0.00 |
| **DACSS-1** | 904.88 | 11.75 | 987.47 | 9.33 | 1053.96 | 10.14 | 1205.52 | 17.11 |
| **DACSS-2** | 898.93 | 11.31 | 988.52 | 8.20 | 1051.98 | 14.75 | 1209.66 | 9.25 |
| **DACSS-3** | 892.34 | 11.58 | 977.58 | 12.16 | 1031.99 | 13.70 | 1197.58 | 17.55 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 11409.00 | 0.00 | 463.00 | 0.00 | 76254.00 | 0.00 | 108289.00 | 0.00 |
| **DACSS-1** | 935.29 | 12.19 | 1075.84 | 29.60 | 31984.13 | 53.13 | 87615.82 | 1321.99 |
| **DACSS-2** | 933.03 | 9.20 | 1073.98 | 60.16 | 32003.46 | 43.01 | 88051.73 | 1877.66 |
| **DACSS-3** | 921.47 | 12.97 | 1068.35 | 32.85 | 31944.15 | 43.25 | 86151.42 | 1416.57 |

## Table 111. Dynamic DACSS Solution Cost Results - File 9 (10 Nodes)

| Solution Cost Results - 10 Node DACSS (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 410.00 | 0.00 | 410.00 | 0.00 | 410.00 | 0.00 |
| **DACSS-1** | 929.41 | 19.33 | 928.14 | 15.22 | 924.96 | 30.61 | 929.46 | 27.36 |
| **DACSS-2** | 925.04 | 47.16 | 929.05 | 14.30 | 935.32 | 21.12 | 934.57 | 15.23 |
| **DACSS-3** | 911.65 | 14.10 | 904.82 | 16.51 | 898.66 | 30.51 | 909.69 | 21.56 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 384.40 | 0.00 | 355.83 | 0.00 | 321.00 | 0.00 |
| **DACSS-1** | 891.19 | 19.24 | 889.30 | 15.32 | 909.60 | 13.39 | 902.06 | 13.50 |
| **DACSS-2** | 894.73 | 17.10 | 893.84 | 16.70 | 908.30 | 12.90 | 903.10 | 11.46 |
| **DACSS-3** | 873.87 | 14.09 | 863.06 | 24.11 | 888.48 | 14.17 | 880.08 | 12.31 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| **MILP** | 410.00 | 0.00 | 430.00 | 0.00 | 24529.80 | 0.00 | 22454.80 | 0.00 |
| **DACSS-1** | 949.52 | 21.52 | 858.40 | 119.58 | 24970.18 | 126.89 | 22966.08 | 30.89 |
| **DACSS-2** | 949.74 | 29.04 | 908.87 | 99.56 | 25045.23 | 114.48 | 22960.65 | 52.35 |
| **DACSS-3** | 939.68 | 21.23 | 861.77 | 124.31 | 24937.37 | 101.57 | 22931.81 | 32.51 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 392.75 | 0.00 | 432.00 | 0.00 | 531.80 | 0.00 |
| **DACSS-1** | 981.74 | 19.39 | 1023.15 | 15.72 | 1051.37 | 27.25 | 1116.16 | 15.68 |
| **DACSS-2** | 982.62 | 24.02 | 1023.66 | 14.85 | 1055.11 | 19.18 | 1108.93 | 31.08 |
| **DACSS-3** | 945.44 | 39.45 | 993.90 | 19.92 | 1032.65 | 31.00 | 1094.47 | 21.30 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 426.00 | 0.00 | 547.25 | 0.00 | 89378.00 | 0.00 |
| **DACSS-1** | 929.50 | 31.12 | 1050.99 | 10.40 | 1121.01 | 11.83 | 47206.62 | 8709.15 |
| **DACSS-2** | 936.85 | 17.66 | 1051.04 | 18.30 | 1126.44 | 11.73 | 50256.84 | 5721.29 |
| **DACSS-3** | 924.16 | 19.32 | 1033.23 | 20.62 | 1107.50 | 17.49 | 35044.48 | 8050.76 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 410.00 | 0.00 | 437.00 | 0.00 | 21401.00 | 0.00 | 92267.00 | 0.00 |
| **DACSS-1** | 937.10 | 16.59 | 1047.94 | 93.66 | 26952.94 | 11.32 | 111062.93 | 2241.26 |
| **DACSS-2** | 925.57 | 27.35 | 1048.73 | 84.79 | 26950.94 | 19.52 | 110562.07 | 1892.24 |
| **DACSS-3** | 918.88 | 30.69 | 1006.53 | 95.79 | 26928.93 | 22.41 | 109878.47 | 914.12 |

Table 112. Dynamic DACSS Solution Cost Results - File 10 (10 Nodes)

| Solution Cost Results - 10 Node DACSS (File 10) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 30328.40 | 0.00 | 30328.40 | 0.00 | 30328.40 | 0.00 |
| **DACSS-1** | 30762.93 | 14.15 | 30754.27 | 14.08 | 30756.69 | 14.78 | 30759.38 | 12.24 |
| **DACSS-2** | 30758.69 | 13.20 | 30756.17 | 12.45 | 30759.18 | 12.13 | 30756.10 | 13.46 |
| **DACSS-3** | 30739.67 | 16.12 | 30743.44 | 11.09 | 30745.61 | 13.82 | 30745.16 | 14.25 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 29334.00 | 0.00 | 26342.60 | 0.00 | 26360.30 | 0.00 |
| **DACSS-1** | 30789.15 | 14.86 | 29773.36 | 12.82 | 26763.33 | 15.79 | 27359.58 | 1038.72 |
| **DACSS-2** | 30786.50 | 8.00 | 29775.28 | 9.50 | 26763.63 | 16.56 | 27296.69 | 1024.50 |
| **DACSS-3** | 30771.65 | 15.66 | 29760.42 | 16.76 | 26758.42 | 12.77 | 26921.08 | 30.13 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| **MILP** | 30328.40 | 0.00 | 377.00 | 0.00 | 479.40 | 0.00 | 52338.00 | 0.00 |
| **DACSS-1** | 30759.98 | 14.28 | 858.55 | 9.66 | 974.11 | 14.03 | 21958.71 | 17.77 |
| **DACSS-2** | 30763.67 | 11.37 | 860.14 | 7.36 | 981.84 | 11.60 | 21965.80 | 12.46 |
| **DACSS-3** | 30746.14 | 11.90 | 847.26 | 13.39 | 971.17 | 17.90 | 21941.90 | 25.72 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 382.40 | 0.00 | 388.00 | 0.00 | 30351.20 | 0.00 |
| **DACSS-1** | 30814.40 | 11.34 | 959.90 | 18.95 | 1010.59 | 19.91 | 1039.30 | 21.95 |
| **DACSS-2** | 30811.93 | 13.76 | 951.43 | 24.42 | 1013.54 | 16.80 | 1042.37 | 19.76 |
| **DACSS-3** | 30796.27 | 14.91 | 943.96 | 18.52 | 990.86 | 21.72 | 1023.00 | 22.46 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 35389.00 | 0.00 | 450.00 | 0.00 | 22495.00 | 0.00 |
| **DACSS-1** | 30807.36 | 11.16 | 35897.33 | 18.33 | 1100.93 | 14.12 | 27900.04 | 683.64 |
| **DACSS-2** | 30811.30 | 11.37 | 35894.11 | 15.10 | 1096.30 | 16.88 | 27978.57 | 605.83 |
| **DACSS-3** | 30790.93 | 13.59 | 35853.35 | 18.50 | 1084.57 | 21.22 | 26904.61 | 959.24 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 30328.40 | 0.00 | 419.67 | 0.00 | 29429.00 | 0.00 | 63365.60 | 0.00 |
| **DACSS-1** | 30853.42 | 14.06 | 1034.74 | 21.49 | 30120.01 | 24.32 | 89455.07 | 5123.22 |
| **DACSS-2** | 30862.96 | 11.18 | 1033.48 | 29.04 | 30128.85 | 26.16 | 88620.75 | 5499.38 |
| **DACSS-3** | 30835.12 | 14.72 | 1005.32 | 54.35 | 30081.41 | 43.04 | 84923.75 | 3185.59 |

## Appendix F: Phase 3 (Distributed) Results - 15 Node Network

### Table 113. Dynamic DACSE Solution Cost Results - File 1 (15 Nodes)

| Solution Cost Results - 15 Node DACSE (File 1) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 |
| **DACSE-1** | 6390.52 | 5724.36 | 8131.50 | 5794.04 | 7327.73 | 6080.74 | 6567.58 | 5595.71 |
| **DACSE-2** | 7785.11 | 5139.34 | 8330.40 | 6492.14 | 7467.86 | 6835.13 | 5977.23 | 4344.30 |
| **DACSE-3** | 5683.38 | 5585.61 | 8634.43 | 6484.31 | 8823.51 | 7401.51 | 7001.37 | 5632.08 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 700.63 | 0.00 | 746.60 | 0.00 | 758.00 | 0.00 |
| **DACSE-1** | 5986.90 | 4609.14 | 5041.79 | 3082.11 | 22659.82 | 11188.46 | 18576.88 | 8984.20 |
| **DACSE-2** | 7531.20 | 6717.28 | 5923.64 | 5299.72 | 21120.22 | 8103.60 | 12433.55 | 10130.17 |
| **DACSE-3** | 7576.52 | 4591.67 | 5717.34 | 6936.81 | 21150.38 | 6501.15 | 16414.10 | 12172.44 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **DACSE-1** | 10080.20 | 6362.74 | 27360.22 | 9234.40 | 15335.86 | 6524.75 | 17355.82 | 17323.49 |
| **DACSE-2** | 7881.58 | 6435.95 | 26875.64 | 8488.53 | 15589.02 | 5520.83 | 16139.52 | 12348.22 |
| **DACSE-3** | 5266.51 | 3943.47 | 23964.93 | 9417.47 | 21287.38 | 9845.16 | 15027.03 | 15193.80 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 777.50 | 0.00 | 879.20 | 0.00 | 49841.00 | 0.00 |
| **DACSE-1** | 7972.93 | 5932.54 | 15317.97 | 8521.91 | 7622.30 | 1069.27 | 17667.06 | 7866.00 |
| **DACSE-2** | 8459.48 | 5153.17 | 12419.51 | 8365.57 | 7743.83 | 433.46 | 16416.26 | 7271.04 |
| **DACSE-3** | 7486.08 | 4739.05 | 11369.56 | 8866.15 | 7462.56 | 4342.89 | 17168.47 | 7968.70 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 39729.80 | 0.00 | 82769.00 | 0.00 | 211614.00 | 0.00 |
| **DACSE-1** | 8756.05 | 6687.31 | 74183.21 | 11824.56 | 130595.47 | 32268.79 | 246344.33 | 9339.04 |
| **DACSE-2** | 9581.71 | 6059.27 | 65926.87 | 11355.81 | 142987.30 | 35395.09 | 247559.50 | 9023.21 |
| **DACSE-3** | 6738.71 | 4720.51 | 70875.13 | 13592.24 | 137136.26 | 32545.73 | 243259.73 | 11803.90 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 796.60 | 0.00 | 1898.00 | 0.00 | 19903.00 | 0.00 |
| **DACSE-1** | 6892.98 | 6260.04 | 20382.45 | 16247.27 | 4663.62 | 2307.99 | 18661.62 | 9111.81 |
| **DACSE-2** | 5641.47 | 4764.72 | 16655.72 | 11618.73 | 4950.06 | 1828.37 | 10230.29 | 6240.22 |
| **DACSE-3** | 7170.96 | 5009.15 | 11551.60 | 8715.47 | 5150.41 | 2323.29 | 23401.83 | 13931.34 |

## Table 114. Dynamic DACSE Solution Cost Results - File 2 (15 Nodes)

| Solution Cost Results - 15 Node DACSE (File 2) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 |
| **DACSE-1** | 11315.33 | 6670.68 | 11307.13 | 10877.62 | 11973.74 | 6620.05 | 15551.58 | 9070.00 |
| **DACSE-2** | 8932.09 | 6335.08 | 12951.63 | 7047.67 | 12477.89 | 7446.00 | 12172.33 | 6150.08 |
| **DACSE-3** | 11033.78 | 6591.08 | 14279.63 | 8867.45 | 11831.04 | 8247.18 | 13131.66 | 8272.70 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 700.63 | 0.00 | 746.60 | 0.00 | 758.00 | 0.00 |
| **DACSE-1** | 12270.55 | 9067.93 | 25865.90 | 9743.31 | 24544.10 | 9570.51 | 10972.48 | 4141.67 |
| **DACSE-2** | 11493.65 | 6339.95 | 23130.31 | 10024.02 | 23260.78 | 9663.59 | 10457.46 | 4593.51 |
| **DACSE-3** | 8504.54 | 6042.62 | 19499.41 | 12201.31 | 21558.13 | 11173.29 | 10618.45 | 6737.40 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **DACSE-1** | 11068.81 | 5778.07 | 24487.01 | 8989.39 | 9619.46 | 7055.41 | 18547.92 | 12220.16 |
| **DACSE-2** | 10949.83 | 6301.05 | 28205.53 | 10492.12 | 11253.84 | 6342.51 | 16840.47 | 10238.58 |
| **DACSE-3** | 11714.45 | 8257.05 | 30066.78 | 10582.77 | 12226.47 | 7043.42 | 21188.77 | 11222.55 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 777.50 | 0.00 | 879.20 | 0.00 | 49841.00 | 0.00 |
| **DACSE-1** | 16440.82 | 8592.40 | 51768.94 | 4244.41 | 60477.15 | 12567.89 | 80754.00 | 15670.08 |
| **DACSE-2** | 12344.61 | 8051.92 | 51945.97 | 3815.34 | 60566.43 | 13373.78 | 70764.64 | 16215.48 |
| **DACSE-3** | 9750.74 | 7026.57 | 55491.97 | 9624.65 | 58682.19 | 9281.26 | 78985.97 | 16169.08 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 39729.80 | 0.00 | 82769.00 | 0.00 | 211614.00 | 0.00 |
| **DACSE-1** | 13155.49 | 7868.91 | 37074.72 | 8739.85 | 72361.34 | 20723.46 | 139716.49 | 24371.81 |
| **DACSE-2** | 11711.98 | 6314.20 | 39378.59 | 10581.88 | 69101.30 | 22542.79 | 150099.77 | 18132.98 |
| **DACSE-3** | 9159.39 | 6596.13 | 38670.01 | 13686.41 | 72386.08 | 13942.40 | 136794.43 | 25726.79 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 796.60 | 0.00 | 1898.00 | 0.00 | 19903.00 | 0.00 |
| **DACSE-1** | 10285.63 | 6775.11 | 12579.99 | 10199.29 | 74771.08 | 7596.49 | 126327.51 | 12605.56 |
| **DACSE-2** | 9714.47 | 5614.63 | 13879.20 | 7828.95 | 87295.01 | 27447.07 | 131556.33 | 8224.80 |
| **DACSE-3** | 11707.72 | 9151.23 | 18071.38 | 12045.14 | 76759.53 | 9497.77 | 121835.09 | 12031.80 |

## Table 115. Dynamic DACSE Solution Cost Results - File 3 (15 Nodes)

| Solution Cost Results - 15 Node DACSE (File 3) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 735.00 | 0.00 | 779.00 | 0.00 | 841.00 | 0.00 |
| **DACSE-1** | 18499.89 | 11989.76 | 18164.61 | 10533.71 | 13125.82 | 9852.18 | 16525.52 | 10113.53 |
| **DACSE-2** | 15786.41 | 10352.64 | 13893.77 | 9585.16 | 13282.85 | 10672.76 | 15911.92 | 8944.67 |
| **DACSE-3** | 12066.85 | 8107.72 | 16842.91 | 11622.98 | 16709.88 | 11388.69 | 13119.72 | 10948.01 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 735.00 | 0.00 | 779.00 | 0.00 | 841.00 | 0.00 |
| **DACSE-1** | 18269.63 | 11173.54 | 38669.57 | 16453.62 | 27404.70 | 9903.08 | 35851.69 | 13838.29 |
| **DACSE-2** | 15342.90 | 9971.78 | 33894.11 | 15269.36 | 28378.00 | 13314.87 | 43798.88 | 18177.92 |
| **DACSE-3** | 14491.19 | 10315.56 | 35128.68 | 14639.11 | 26944.02 | 11075.75 | 34758.31 | 15553.33 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 855.80 | 0.00 | 975.87 | 0.00 | 982.13 | 0.00 |
| **DACSE-1** | 15652.74 | 10766.27 | 45025.70 | 9070.59 | 74956.68 | 9080.92 | 109323.44 | 16547.91 |
| **DACSE-2** | 14309.92 | 8864.21 | 40099.40 | 15151.58 | 77441.70 | 10244.02 | 99302.29 | 20000.55 |
| **DACSE-3** | 17731.89 | 12061.85 | 37355.53 | 13293.24 | 68048.44 | 14271.90 | 101689.16 | 17271.16 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 768.65 | 0.00 | 830.70 | 0.00 | 954.80 | 0.00 |
| **DACSE-1** | 14084.34 | 7917.35 | 15744.68 | 12095.39 | 30145.02 | 14842.09 | 16209.87 | 4768.22 |
| **DACSE-2** | 19050.45 | 14097.44 | 16371.25 | 12639.06 | 24790.84 | 16259.11 | 13494.66 | 2193.35 |
| **DACSE-3** | 16155.00 | 9755.62 | 17533.14 | 11093.19 | 19428.92 | 10844.16 | 18157.81 | 4707.19 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 893.20 | 0.00 | 1127.83 | 0.00 | 115904.00 | 0.00 |
| **DACSE-1** | 18746.14 | 12840.29 | 36910.29 | 15746.21 | 51299.73 | 11430.93 | 134201.10 | 19663.78 |
| **DACSE-2** | 15551.58 | 10064.32 | 37531.62 | 13497.74 | 45716.92 | 10238.81 | 144830.47 | 15518.56 |
| **DACSE-3** | 12532.89 | 7848.90 | 35375.80 | 16044.93 | 51479.60 | 9855.57 | 131365.43 | 17828.04 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 771.00 | 0.00 | 39913.00 | 0.00 | 39023.80 | 0.00 |
| **DACSE-1** | 17582.60 | 12516.63 | 17474.27 | 13494.39 | 67934.44 | 13912.98 | 128747.00 | 14898.38 |
| **DACSE-2** | 13405.63 | 9521.54 | 14806.17 | 10559.95 | 73600.25 | 11418.67 | 134012.07 | 13368.56 |
| **DACSE-3** | 17087.84 | 11411.39 | 21176.45 | 15709.07 | 66767.12 | 13568.17 | 136811.73 | 18095.65 |

**Table 116. Dynamic DACSE Solution Cost Results - File 4 (15 Nodes)**

| Solution Cost Results - 15 Node DACSE (File 4) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 42836.00 | 0.00 | 42836.00 | 0.00 | 42836.00 | 0.00 |
| **DACSE-1** | 69595.26 | 10637.68 | 66368.63 | 5915.01 | 65192.10 | 5486.74 | 65496.23 | 4697.06 |
| **DACSE-2** | 70368.45 | 14784.13 | 72095.12 | 15411.47 | 66782.08 | 6330.98 | 71755.90 | 12222.91 |
| **DACSE-3** | 74124.33 | 15049.35 | 64782.23 | 7272.14 | 66760.41 | 8131.92 | 64149.83 | 7045.25 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 52838.50 | 0.00 | 51858.80 | 0.00 | 51833.00 | 0.00 |
| **DACSE-1** | 69949.17 | 14921.86 | 79340.80 | 17683.36 | 62760.78 | 6762.60 | 73205.89 | 7489.44 |
| **DACSE-2** | 75761.30 | 15530.18 | 80508.10 | 18633.92 | 64169.88 | 8957.71 | 74739.34 | 11478.93 |
| **DACSE-3** | 71818.75 | 12578.83 | 69832.09 | 10985.23 | 65967.58 | 9455.78 | 76008.97 | 10005.59 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 42869.30 | 0.00 | 987.53 | 0.00 | 1012.00 | 0.00 |
| **DACSE-1** | 70450.07 | 13256.43 | 54464.45 | 8718.68 | 85438.82 | 26165.42 | 47354.62 | 22341.53 |
| **DACSE-2** | 74523.81 | 14104.43 | 58959.15 | 11499.29 | 92612.85 | 26492.21 | 55857.74 | 18732.20 |
| **DACSE-3** | 69657.36 | 11870.62 | 56125.28 | 10558.73 | 81288.23 | 26867.84 | 39518.48 | 22518.78 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 85712.00 | 0.00 | 76762.40 | 0.00 | 95755.50 | 0.00 |
| **DACSE-1** | 69187.96 | 8539.24 | 111791.42 | 10288.50 | 90849.67 | 10604.40 | 126224.47 | 10373.33 |
| **DACSE-2** | 70609.40 | 11920.98 | 118234.15 | 12061.12 | 90863.24 | 6356.69 | 126368.77 | 9899.21 |
| **DACSE-3** | 71084.54 | 13379.84 | 113812.34 | 9621.46 | 93305.11 | 12628.56 | 129559.67 | 14270.91 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 904.00 | 0.00 | 861.33 | 0.00 | 77844.50 | 0.00 |
| **DACSE-1** | 70322.95 | 12242.69 | 29312.75 | 11698.60 | 20386.89 | 15253.47 | 173367.47 | 25498.65 |
| **DACSE-2** | 69956.55 | 12827.69 | 31666.77 | 15583.39 | 25866.41 | 10645.30 | 174019.20 | 20217.04 |
| **DACSE-3** | 70866.25 | 13103.64 | 31447.91 | 15320.99 | 19722.57 | 11909.74 | 141221.19 | 36316.40 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 817.80 | 0.00 | 46979.70 | 0.00 | 75918.00 | 0.00 |
| **DACSE-1** | 72064.95 | 14878.73 | 5748.26 | 4148.12 | 92016.81 | 14184.84 | 253952.00 | 13521.26 |
| **DACSE-2** | 75199.19 | 14489.04 | 7991.94 | 5260.54 | 89163.87 | 10845.23 | 245507.20 | 13033.59 |
| **DACSE-3** | 73672.20 | 18436.41 | 13971.22 | 8765.68 | 88295.51 | 12806.22 | 251632.00 | 15019.54 |

Table 117. Dynamic DACSE Solution Cost Results - File 5 (15 Nodes)

| Solution Cost Results - 15 Node DACSE (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 747.00 | 0.00 | 747.00 | 0.00 | 747.00 | 0.00 |
| **DACSE-1** | 9941.90 | 7197.82 | 9107.91 | 8358.32 | 9669.05 | 7357.88 | 8803.61 | 7221.75 |
| **DACSE-2** | 10717.65 | 7538.26 | 7809.90 | 6076.03 | 11693.65 | 8681.00 | 9262.30 | 6694.51 |
| **DACSE-3** | 8696.94 | 7157.09 | 8989.82 | 7465.16 | 12178.58 | 17591.39 | 13347.82 | 19469.35 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 776.60 | 0.00 | 790.00 | 0.00 | 799.25 | 0.00 |
| **DACSE-1** | 12151.63 | 7514.35 | 12369.60 | 5002.90 | 8888.53 | 5112.20 | 8789.62 | 3951.98 |
| **DACSE-2** | 10495.07 | 6808.03 | 13899.56 | 6236.66 | 9519.02 | 7014.67 | 12075.75 | 5060.21 |
| **DACSE-3** | 12222.93 | 17206.05 | 12305.14 | 7506.84 | 8637.89 | 5338.65 | 12473.91 | 6400.71 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 775.33 | 0.00 | 970.40 | 0.00 | 43834.20 | 0.00 |
| **DACSE-1** | 8994.32 | 6523.21 | 34735.46 | 13600.72 | 13206.56 | 12158.66 | 87563.92 | 23367.57 |
| **DACSE-2** | 11235.17 | 6655.18 | 34450.60 | 11885.51 | 14288.13 | 11814.39 | 80741.02 | 17340.59 |
| **DACSE-3** | 10446.80 | 7688.34 | 29258.50 | 13986.91 | 24917.84 | 14741.85 | 83185.81 | 17399.76 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 749.92 | 0.00 | 764.00 | 0.00 | 794.40 | 0.00 |
| **DACSE-1** | 10741.38 | 7178.33 | 17386.12 | 10237.89 | 82459.71 | 17796.26 | 51495.57 | 14148.42 |
| **DACSE-2** | 10460.87 | 7304.86 | 16113.50 | 9830.76 | 77064.78 | 17359.35 | 51194.98 | 12254.21 |
| **DACSE-3** | 11811.27 | 8994.46 | 23717.31 | 14171.52 | 76738.75 | 19528.25 | 53094.56 | 16493.55 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 768.40 | 0.00 | 832.00 | 0.00 | 1009.88 | 0.00 |
| **DACSE-1** | 11524.93 | 7065.03 | 7320.67 | 6155.22 | 6693.03 | 5166.61 | 92978.44 | 24316.04 |
| **DACSE-2** | 8956.14 | 8118.89 | 11873.06 | 7144.28 | 3650.87 | 1753.31 | 116465.26 | 21737.23 |
| **DACSE-3** | 11337.47 | 5536.21 | 8830.27 | 6426.07 | 6563.12 | 4969.26 | 87839.04 | 25301.77 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 765.50 | 0.00 | 49890.00 | 0.00 | 126821.00 | 0.00 |
| **DACSE-1** | 8983.42 | 8252.27 | 41555.67 | 6445.89 | 93950.33 | 30526.40 | 248590.67 | 8392.29 |
| **DACSE-2** | 8576.94 | 7134.91 | 40986.85 | 6772.04 | 94759.19 | 15549.45 | 241873.90 | 12142.32 |
| **DACSE-3** | 9301.57 | 5495.71 | 41363.73 | 9035.57 | 87918.48 | 24087.89 | 247962.07 | 8795.03 |

## Table 118. Dynamic DACSE Solution Cost Results - File 6 (15 Nodes)

| Solution Cost Results - 15 Node DACSE (File 6) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 811.25 | 0.00 | 811.25 | 0.00 | 811.25 | 0.00 |
| **DACSE-1** | 16732.60 | 10598.03 | 17894.93 | 11618.42 | 17005.26 | 7982.97 | 18047.32 | 11860.14 |
| **DACSE-2** | 19838.89 | 12331.40 | 19569.16 | 13126.33 | 18208.96 | 10678.63 | 19635.17 | 11856.25 |
| **DACSE-3** | 19278.92 | 14289.51 | 15795.90 | 10022.21 | 21902.68 | 11009.38 | 18059.43 | 10573.60 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 879.00 | 0.00 | 1022.02 | 0.00 | 1107.87 | 0.00 |
| **DACSE-1** | 19011.41 | 13665.21 | 33490.87 | 8221.02 | 21273.77 | 11271.96 | 57281.05 | 17308.58 |
| **DACSE-2** | 21581.76 | 13845.10 | 35161.01 | 10982.36 | 36364.05 | 13568.83 | 59344.66 | 12676.14 |
| **DACSE-3** | 22674.44 | 13843.39 | 31055.19 | 13650.53 | 25645.74 | 16172.30 | 55560.00 | 20862.23 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 853.15 | 0.00 | 55158.60 | 0.00 | 50920.20 | 0.00 |
| **DACSE-1** | 21172.20 | 14419.33 | 34709.06 | 7045.78 | 57189.58 | 17628.45 | 47481.90 | 10280.38 |
| **DACSE-2** | 19549.71 | 12326.07 | 29617.25 | 6935.58 | 64853.49 | 16735.50 | 51642.58 | 8833.00 |
| **DACSE-3** | 20603.40 | 14124.49 | 32215.70 | 13628.75 | 56083.44 | 17000.99 | 51510.22 | 13463.09 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 772.00 | 0.00 | 13766.80 | 0.00 | 13790.00 | 0.00 |
| **DACSE-1** | 21486.74 | 17077.86 | 15291.61 | 12310.16 | 28550.98 | 10035.21 | 53277.99 | 11411.25 |
| **DACSE-2** | 20666.01 | 13818.08 | 22451.63 | 15938.36 | 25175.24 | 5660.81 | 53004.84 | 9331.00 |
| **DACSE-3** | 22166.93 | 11554.99 | 17645.70 | 9448.19 | 29189.76 | 9441.77 | 50021.11 | 12081.76 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 898.00 | 0.00 | 888.50 | 0.00 | 2960.33 | 0.00 |
| **DACSE-1** | 19584.60 | 12946.72 | 41093.70 | 13636.95 | 23888.33 | 12259.20 | 32743.14 | 23202.69 |
| **DACSE-2** | 21100.47 | 12765.13 | 32727.07 | 10542.97 | 25936.03 | 12604.80 | 31654.48 | 24326.98 |
| **DACSE-3** | 22831.50 | 14025.11 | 35114.74 | 14843.26 | 22954.82 | 10120.35 | 20766.48 | 13105.07 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 833.00 | 0.00 | 908.35 | 0.00 | 133843.00 | 0.00 |
| **DACSE-1** | 26522.32 | 14422.77 | 32462.26 | 21474.71 | 25618.45 | 13782.84 | 223483.67 | 55141.01 |
| **DACSE-2** | 19407.59 | 11512.24 | 23946.76 | 14067.97 | 27781.88 | 9764.90 | 225013.03 | 32967.40 |
| **DACSE-3** | 22926.13 | 14835.11 | 27792.59 | 14985.55 | 24424.55 | 8906.33 | 212282.10 | 45874.71 |

Table 119. Dynamic DACSE Solution Cost Results - File 7 (15 Nodes)

| Solution Cost Results - 15 Node DACSE (File 7) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 829.80 | 0.00 | 829.80 | 0.00 | 829.80 | 0.00 |
| **DACSE-1** | 11183.69 | 5945.55 | 12283.65 | 6888.08 | 11955.70 | 9233.93 | 12063.14 | 7651.94 |
| **DACSE-2** | 13705.08 | 7978.30 | 12840.79 | 6944.19 | 11866.05 | 4811.93 | 11671.51 | 4888.03 |
| **DACSE-3** | 12346.18 | 4756.67 | 16673.97 | 11665.34 | 15592.44 | 7617.65 | 15135.23 | 8602.36 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 856.00 | 0.00 | 834.40 | 0.00 | 818.90 | 0.00 |
| **DACSE-1** | 13263.97 | 7009.87 | 17893.89 | 12815.19 | 14206.23 | 9581.60 | 20483.94 | 8740.80 |
| **DACSE-2** | 12988.71 | 5514.40 | 11478.46 | 10808.56 | 16917.38 | 11342.31 | 24460.71 | 12305.47 |
| **DACSE-3** | 16559.33 | 8042.48 | 13854.61 | 10207.04 | 9218.47 | 7757.38 | 21169.26 | 10144.99 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 838.00 | 0.00 | 1168.85 | 0.00 | 1084.87 | 0.00 |
| **DACSE-1** | 13239.46 | 7303.21 | 23492.94 | 11100.46 | 45308.36 | 14933.34 | 37042.32 | 6764.17 |
| **DACSE-2** | 13875.88 | 4916.70 | 24863.46 | 7267.44 | 36936.89 | 16213.84 | 37740.18 | 6313.25 |
| **DACSE-3** | 14716.47 | 8643.83 | 20249.80 | 11942.86 | 41695.86 | 16000.91 | 35392.84 | 8842.09 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 784.00 | 0.00 | 901.00 | 0.00 | 936.20 | 0.00 |
| **DACSE-1** | 12501.85 | 6518.12 | 36609.50 | 18725.84 | 104485.05 | 21577.13 | 74584.54 | 18681.07 |
| **DACSE-2** | 11113.97 | 4589.97 | 28764.03 | 18376.81 | 96385.33 | 16467.04 | 95367.41 | 20705.30 |
| **DACSE-3** | 13840.62 | 7372.17 | 28946.79 | 17935.97 | 95769.03 | 17082.88 | 73729.76 | 18027.64 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 864.67 | 0.00 | 895.00 | 0.00 | 52112.20 | 0.00 |
| **DACSE-1** | 10816.78 | 4673.84 | 29066.72 | 11441.48 | 45626.33 | 11496.95 | 34979.61 | 15877.41 |
| **DACSE-2** | 11391.68 | 5338.32 | 29004.48 | 9800.18 | 49091.38 | 11028.32 | 37106.22 | 6903.11 |
| **DACSE-3** | 12593.40 | 6659.46 | 23628.15 | 9682.99 | 43550.82 | 14293.84 | 38808.91 | 10019.41 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 900.80 | 0.00 | 807.93 | 0.00 | 985.75 | 0.00 |
| **DACSE-1** | 12725.21 | 8160.54 | 60521.13 | 24157.54 | 35646.95 | 16171.37 | 32685.15 | 6996.54 |
| **DACSE-2** | 12090.04 | 6395.64 | 56830.40 | 18436.62 | 28124.29 | 11552.91 | 29223.29 | 4486.19 |
| **DACSE-3** | 9038.32 | 6180.27 | 48052.21 | 16695.62 | 32159.91 | 15464.27 | 32204.11 | 5616.70 |

**Table 120. Dynamic DACSE Solution Cost Results - File 8 (15 Nodes)**

| Solution Cost Results - 15 Node DACSE (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 788.25 | 0.00 | 788.25 | 0.00 | 788.25 | 0.00 |
| DACSE-1 | 29891.49 | 22110.80 | 32009.40 | 16484.03 | 24527.72 | 16891.16 | 27748.54 | 14848.47 |
| DACSE-2 | 22067.68 | 14961.46 | 29411.74 | 24118.55 | 22402.04 | 12194.97 | 30122.75 | 23248.40 |
| DACSE-3 | 26904.56 | 19863.88 | 23589.52 | 16854.63 | 21761.68 | 17616.46 | 27920.75 | 20872.65 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 840.75 | 0.00 | 933.70 | 0.00 | 925.17 | 0.00 |
| DACSE-1 | 31139.85 | 21970.78 | 17904.27 | 8374.67 | 26968.21 | 15773.05 | 12937.31 | 2479.12 |
| DACSE-2 | 28985.88 | 24386.25 | 21887.95 | 9918.22 | 23500.84 | 12320.12 | 15957.64 | 10709.38 |
| DACSE-3 | 29591.24 | 23526.31 | 16641.43 | 9714.16 | 20551.78 | 14526.39 | 17004.99 | 10207.90 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 825.00 | 0.00 | 48864.50 | 0.00 | 77794.00 | 0.00 |
| DACSE-1 | 28797.15 | 17288.90 | 48078.66 | 24747.23 | 74027.33 | 14783.94 | 87863.18 | 9280.20 |
| DACSE-2 | 26325.71 | 20929.38 | 32504.24 | 22222.17 | 70648.32 | 17451.34 | 87501.00 | 11188.44 |
| DACSE-3 | 27480.73 | 18543.72 | 37894.29 | 15213.17 | 73414.50 | 15791.13 | 86510.95 | 14029.21 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 836.25 | 0.00 | 44774.60 | 0.00 | 45835.40 | 0.00 |
| DACSE-1 | 29409.41 | 19026.22 | 9555.44 | 6541.72 | 75290.60 | 15677.41 | 88599.22 | 13021.00 |
| DACSE-2 | 28344.92 | 19121.40 | 10222.60 | 10144.61 | 66848.51 | 11328.15 | 82205.77 | 18341.08 |
| DACSE-3 | 25751.26 | 15995.28 | 11609.37 | 6982.92 | 74259.04 | 15188.14 | 87435.65 | 14998.47 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 764.45 | 0.00 | 833.20 | 0.00 | 58801.20 | 0.00 |
| DACSE-1 | 26981.85 | 16000.31 | 15334.11 | 8259.54 | 54055.49 | 16425.32 | 97826.87 | 12565.49 |
| DACSE-2 | 26320.26 | 17470.44 | 15839.57 | 9812.16 | 65308.51 | 19527.43 | 93797.57 | 6430.48 |
| DACSE-3 | 29577.65 | 20321.98 | 15279.72 | 9895.29 | 49196.79 | 19403.54 | 97661.37 | 9871.95 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 788.25 | 0.00 | 925.20 | 0.00 | 15849.10 | 0.00 | 1928.50 | 0.00 |
| DACSE-1 | 31647.06 | 19111.94 | 34570.16 | 16905.60 | 57371.06 | 15859.21 | 144224.03 | 15089.23 |
| DACSE-2 | 31558.34 | 20048.63 | 38869.33 | 24486.49 | 68066.15 | 11743.59 | 134465.20 | 1718.04 |
| DACSE-3 | 35670.10 | 21702.96 | 32928.62 | 18028.00 | 53863.77 | 23866.61 | 138042.87 | 7068.28 |

## Table 121. Dynamic DACSE Solution Cost Results - File 9 (15 Nodes)

| Solution Cost Results - 15 Node DACSE (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 730.60 | 0.00 | 730.60 | 0.00 | 730.60 | 0.00 |
| **DACSE-1** | 5092.38 | 3764.31 | 7180.20 | 6022.66 | 6655.02 | 4646.82 | 11912.90 | 11103.08 |
| **DACSE-2** | 6031.91 | 6686.23 | 5833.94 | 4674.75 | 7607.26 | 7373.42 | 6979.31 | 9808.16 |
| **DACSE-3** | 5347.85 | 4851.33 | 4908.28 | 4583.36 | 8703.91 | 8129.80 | 7217.72 | 6190.35 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 816.00 | 0.00 | 865.25 | 0.00 | 856.75 | 0.00 |
| **DACSE-1** | 7886.69 | 8081.55 | 26066.75 | 25945.37 | 58923.60 | 7921.32 | 52217.48 | 16054.60 |
| **DACSE-2** | 5043.82 | 3379.84 | 38203.93 | 23303.10 | 57477.04 | 3633.40 | 45872.29 | 14942.25 |
| **DACSE-3** | 6486.81 | 5005.90 | 36668.41 | 21071.71 | 58320.89 | 12765.17 | 57760.65 | 16406.02 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 755.67 | 0.00 | 741.00 | 0.00 | 2845.90 | 0.00 |
| **DACSE-1** | 7046.79 | 9957.05 | 7112.96 | 6692.84 | 9951.36 | 5936.97 | 32793.23 | 12894.36 |
| **DACSE-2** | 9086.27 | 13010.92 | 7939.77 | 5472.59 | 8729.96 | 5546.40 | 32501.34 | 13367.98 |
| **DACSE-3** | 8294.61 | 12012.96 | 11026.05 | 10298.88 | 11096.07 | 7352.20 | 22198.43 | 13143.92 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 748.92 | 0.00 | 806.20 | 0.00 | 868.00 | 0.00 |
| **DACSE-1** | 9965.20 | 12245.36 | 33167.43 | 10606.59 | 32457.17 | 10442.06 | 50269.07 | 16185.10 |
| **DACSE-2** | 10814.92 | 14442.04 | 37096.16 | 14120.36 | 35810.07 | 13544.55 | 62705.58 | 16041.83 |
| **DACSE-3** | 7049.74 | 5509.50 | 33634.39 | 11136.72 | 34685.10 | 11784.87 | 54741.46 | 23632.64 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 818.25 | 0.00 | 816.20 | 0.00 | 24820.10 | 0.00 |
| **DACSE-1** | 5947.76 | 4381.97 | 8802.50 | 7590.54 | 5942.08 | 6536.49 | 48190.43 | 7702.30 |
| **DACSE-2** | 6312.98 | 5067.61 | 9628.89 | 7293.84 | 6322.65 | 5870.95 | 48789.13 | 4677.65 |
| **DACSE-3** | 8777.10 | 7846.12 | 16286.75 | 16029.16 | 16948.80 | 12394.16 | 49746.03 | 9477.17 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 851.00 | 0.00 | 950.30 | 0.00 | 117848.00 | 0.00 |
| **DACSE-1** | 5550.23 | 6842.24 | 20660.79 | 7013.31 | 50408.73 | 13581.10 | 233916.53 | 17809.14 |
| **DACSE-2** | 7495.84 | 9020.71 | 36120.49 | 34492.86 | 50705.93 | 8714.05 | 235663.97 | 29452.21 |
| **DACSE-3** | 6607.15 | 5213.31 | 21157.01 | 7874.49 | 50865.26 | 12064.61 | 220217.57 | 27559.84 |

**Table 122. Dynamic DACSE Solution Cost Results - File 10 (15 Nodes)**

| Solution Cost Results - 15 Node DACSE (File 10) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 745.00 | 0.00 | 745.00 | 0.00 | 745.00 | 0.00 | 745.00 | 0.00 |
| DACSE-1 | 9004.26 | 6979.69 | 10648.70 | 7159.53 | 8379.15 | 7632.43 | 8401.59 | 5574.29 |
| DACSE-2 | 7983.44 | 7908.44 | 12497.67 | 16044.11 | 7867.26 | 6382.91 | 7379.49 | 5781.15 |
| DACSE-3 | 11353.10 | 7881.97 | 6927.49 | 4794.04 | 8618.03 | 6881.58 | 11335.56 | 9948.22 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 745.00 | 0.00 | 755.80 | 0.00 | 773.60 | 0.00 | 762.05 | 0.00 |
| DACSE-1 | 9943.66 | 8474.20 | 21038.02 | 10183.07 | 20476.71 | 13003.86 | 7811.70 | 4365.99 |
| DACSE-2 | 9302.78 | 7123.23 | 22634.28 | 10879.59 | 20013.15 | 12902.07 | 7572.07 | 4134.11 |
| DACSE-3 | 11805.60 | 6824.38 | 15886.67 | 9218.61 | 18715.16 | 12183.98 | 11111.39 | 7815.36 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 745.00 | 0.00 | 750.00 | 0.00 | 752.50 | 0.00 | 736.20 | 0.00 |
| DACSE-1 | 8563.18 | 6652.14 | 9458.23 | 5828.66 | 5419.98 | 4801.09 | 17048.24 | 9330.78 |
| DACSE-2 | 7786.32 | 5743.47 | 9199.52 | 8146.91 | 6712.15 | 5955.17 | 13533.96 | 8417.42 |
| DACSE-3 | 8498.33 | 5463.57 | 9709.08 | 7237.93 | 6658.47 | 6135.59 | 12513.73 | 7660.68 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 745.00 | 0.00 | 813.70 | 0.00 | 43833.70 | 0.00 | 31976.20 | 0.00 |
| DACSE-1 | 10029.21 | 7032.48 | 35450.38 | 10115.75 | 49148.27 | 6580.40 | 81861.76 | 15826.02 |
| DACSE-2 | 9448.27 | 6755.73 | 34134.44 | 11077.16 | 46584.93 | 7166.12 | 73441.13 | 11288.40 |
| DACSE-3 | 11981.39 | 7320.54 | 34426.92 | 16229.41 | 50461.58 | 9490.35 | 80275.26 | 16222.74 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 745.00 | 0.00 | 76713.40 | 0.00 | 42806.70 | 0.00 | 20917.00 | 0.00 |
| DACSE-1 | 8298.81 | 6199.52 | 61185.94 | 13038.79 | 83265.24 | 12979.67 | 63326.86 | 18293.10 |
| DACSE-2 | 8272.71 | 5034.26 | 62979.05 | 18701.25 | 81046.34 | 7722.16 | 66530.98 | 15987.34 |
| DACSE-3 | 9370.94 | 6631.11 | 60080.36 | 16225.55 | 90262.95 | 20660.31 | 71861.15 | 17057.18 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 745.00 | 0.00 | 886.20 | 0.00 | 83848.90 | 0.00 | 10029.20 | 0.00 |
| DACSE-1 | 10592.70 | 6808.11 | 58529.42 | 17058.58 | 115422.28 | 16916.56 | 130834.80 | 26030.97 |
| DACSE-2 | 7168.31 | 6486.41 | 50311.96 | 19858.73 | 125023.37 | 20080.08 | 109778.87 | 27589.40 |
| DACSE-3 | 9701.63 | 6699.63 | 53709.13 | 21215.63 | 116831.71 | 19923.32 | 115692.40 | 30465.89 |

**Table 123. Dynamic DACSS Solution Cost Results - File 1 (15 Nodes)**

| Solution Cost Results - 15 Node DACSS (File 1) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 |
| DACSS-1 | 2389.84 | 52.37 | 2394.65 | 44.00 | 2389.58 | 10.98 | 2360.02 | 65.71 |
| DACSS-2 | 2423.75 | 23.58 | 2396.09 | 33.86 | 2408.55 | 38.68 | 2391.12 | 69.73 |
| DACSS-3 | 2305.54 | 73.54 | 2313.89 | 52.57 | 2310.15 | 36.26 | 2334.30 | 36.45 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 700.63 | 0.00 | 746.60 | 0.00 | 758.00 | 0.00 |
| DACSS-1 | 2502.48 | 43.18 | 2508.99 | 45.59 | 2696.80 | 112.29 | 2582.40 | 90.12 |
| DACSS-2 | 2509.17 | 29.06 | 2476.49 | 59.11 | 2658.21 | 93.54 | 2625.85 | 55.23 |
| DACSS-3 | 2360.14 | 67.32 | 2301.20 | 86.95 | 2542.25 | 50.65 | 2513.52 | 21.80 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 674.50 | 0.00 | 710.33 | 0.00 | 746.20 | 0.00 |
| DACSS-1 | 2496.66 | 24.47 | 2420.57 | 16.20 | 2468.31 | 57.34 | 2629.78 | 19.71 |
| DACSS-2 | 2436.87 | 38.28 | 2403.73 | 44.82 | 2527.56 | 67.37 | 2606.53 | 44.38 |
| DACSS-3 | 2360.15 | 44.16 | 2341.11 | 46.40 | 2471.04 | 49.45 | 2581.31 | 69.96 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 777.50 | 0.00 | 879.20 | 0.00 | 49841.00 | 0.00 |
| DACSS-1 | 2468.36 | 45.12 | 2504.72 | 65.34 | 2552.67 | 17.22 | 2922.41 | 66.03 |
| DACSS-2 | 2435.62 | 43.49 | 2508.65 | 47.52 | 2556.01 | 22.53 | 3250.14 | 925.67 |
| DACSS-3 | 2353.32 | 34.69 | 2464.75 | 18.20 | 2500.82 | 22.15 | 2810.16 | 105.22 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 39729.80 | 0.00 | 82769.00 | 0.00 | 211614.00 | 0.00 |
| DACSS-1 | 2542.06 | 48.11 | 41460.87 | 98.29 | 87940.05 | 4320.17 | 211531.20 | 2287.56 |
| DACSS-2 | 2528.56 | 38.43 | 42261.21 | 1650.04 | 89371.31 | 1186.48 | 212337.60 | 2333.03 |
| DACSS-3 | 2434.62 | 45.31 | 41369.58 | 77.24 | 87062.89 | 5492.87 | 210221.30 | 2164.41 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | | Cost/Std Dev | |
| MILP | 696.40 | 0.00 | 796.60 | 0.00 | 1898.00 | 0.00 | 19903.00 | 0.00 |
| DACSS-1 | 2458.59 | 34.17 | 2580.00 | 42.11 | 2558.01 | 84.27 | 6062.83 | 1165.89 |
| DACSS-2 | 2400.84 | 185.07 | 2536.12 | 148.72 | 2491.28 | 65.90 | 3949.19 | 1498.45 |
| DACSS-3 | 2347.98 | 38.04 | 2504.68 | 45.73 | 2571.06 | 43.49 | 3275.95 | 957.74 |

**Table 124. Dynamic DACSS Solution Cost Results - File 2 (15 Nodes)**

| Solution Cost Results - 15 Node DACSS (File 2) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 | 696.40 | 0.00 |
| **DACSS-1** | 2281.62 | 89.42 | 2368.46 | 117.88 | 2334.08 | 97.08 | 2394.39 | 48.53 |
| **DACSS-2** | 2385.53 | 91.99 | 2345.34 | 96.12 | 2347.02 | 64.26 | 2414.02 | 97.26 |
| **DACSS-3** | 2324.72 | 54.04 | 2142.13 | 222.67 | 2272.90 | 120.44 | 2297.66 | 106.07 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 700.63 | 0.00 | 746.60 | 0.00 | 758.00 | 0.00 |
| **DACSS-1** | 2458.60 | 54.01 | 2506.63 | 44.58 | 2681.15 | 28.80 | 2604.61 | 35.05 |
| **DACSS-2** | 2405.73 | 105.43 | 2501.40 | 105.43 | 2635.30 | 222.97 | 2600.47 | 22.81 |
| **DACSS-3** | 2398.83 | 78.36 | 2395.81 | 27.33 | 2634.35 | 48.49 | 2511.71 | 70.43 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 674.50 | 0.00 | 710.33 | 0.00 | 746.20 | 0.00 |
| **DACSS-1** | 2427.65 | 99.98 | 2795.96 | 62.83 | 2510.13 | 27.22 | 2564.50 | 119.54 |
| **DACSS-2** | 2296.03 | 68.97 | 2717.30 | 117.23 | 2554.53 | 30.61 | 2595.17 | 83.28 |
| **DACSS-3** | 2295.07 | 139.44 | 2631.93 | 84.09 | 2479.04 | 46.61 | 2468.76 | 52.33 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 777.50 | 0.00 | 879.20 | 0.00 | 49841.00 | 0.00 |
| **DACSS-1** | 2486.86 | 63.04 | 47491.75 | 47.56 | 52496.12 | 43.14 | 49574.28 | 81.88 |
| **DACSS-2** | 2437.84 | 132.94 | 47517.01 | 33.48 | 52459.92 | 56.00 | 49527.35 | 101.19 |
| **DACSS-3** | 2373.60 | 284.20 | 47409.20 | 57.24 | 52382.86 | 30.08 | 49489.56 | 44.35 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 39729.80 | 0.00 | 82769.00 | 0.00 | 211614.00 | 0.00 |
| **DACSS-1** | 2400.54 | 108.90 | 2608.53 | 84.64 | 7458.18 | 2921.06 | 73775.38 | 8663.74 |
| **DACSS-2** | 2444.15 | 86.06 | 2662.14 | 66.39 | 9603.27 | 2841.91 | 81158.26 | 4264.34 |
| **DACSS-3** | 2391.82 | 66.51 | 2590.65 | 89.93 | 6870.92 | 4155.53 | 75387.57 | 6985.20 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 696.40 | 0.00 | 796.60 | 0.00 | 1898.00 | 0.00 | 19903.00 | 0.00 |
| **DACSS-1** | 2414.80 | 225.56 | 2748.11 | 62.04 | 59562.85 | 1344.71 | 81719.90 | 2389.41 |
| **DACSS-2** | 2533.61 | 59.19 | 2710.57 | 71.12 | 58776.52 | 2078.39 | 82424.76 | 3048.01 |
| **DACSS-3** | 2431.26 | 95.44 | 2633.60 | 60.78 | 59681.70 | 1649.68 | 76087.34 | 5939.66 |

**Table 125. Dynamic DACSS Solution Cost Results - File 3 (15 Nodes)**

| Solution Cost Results - 15 Node DACSS (File 3) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 735.00 | 0.00 | 779.00 | 0.00 | 841.00 | 0.00 |
| **DACSS-1** | 2305.52 | 30.75 | 2314.34 | 27.67 | 2301.71 | 46.90 | 2281.89 | 25.71 |
| **DACSS-2** | 2343.23 | 12.95 | 2301.80 | 23.92 | 2347.37 | 15.54 | 2311.21 | 31.91 |
| **DACSS-3** | 2264.98 | 34.67 | 2293.24 | 44.80 | 2296.99 | 34.68 | 2276.44 | 20.26 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 735.00 | 0.00 | 779.00 | 0.00 | 841.00 | 0.00 |
| **DACSS-1** | 2405.85 | 46.68 | 2778.81 | 676.54 | 2670.31 | 135.71 | 4172.49 | 2159.02 |
| **DACSS-2** | 2383.43 | 72.97 | 2862.54 | 930.57 | 2743.65 | 99.20 | 3662.50 | 1912.76 |
| **DACSS-3** | 2391.15 | 17.70 | 2490.93 | 37.68 | 2649.67 | 89.44 | 3171.34 | 1286.35 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 855.80 | 0.00 | 975.87 | 0.00 | 982.13 | 0.00 |
| **DACSS-1** | 2317.70 | 49.65 | 2459.03 | 62.24 | 7172.45 | 4161.85 | 23726.55 | 9290.32 |
| **DACSS-2** | 2334.21 | 32.27 | 2463.48 | 66.08 | 10007.09 | 5517.60 | 25346.58 | 5061.09 |
| **DACSS-3** | 2290.84 | 27.75 | 2435.16 | 65.16 | 2986.41 | 1460.34 | 19404.76 | 5918.80 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 768.65 | 0.00 | 830.70 | 0.00 | 954.80 | 0.00 |
| **DACSS-1** | 2323.15 | 114.74 | 2717.27 | 44.88 | 2421.78 | 117.63 | 4890.49 | 2626.57 |
| **DACSS-2** | 2403.15 | 28.14 | 2677.26 | 74.35 | 2428.59 | 145.30 | 3770.78 | 1811.63 |
| **DACSS-3** | 2373.12 | 47.23 | 2660.92 | 83.32 | 2527.19 | 123.61 | 3363.52 | 1617.69 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 893.20 | 0.00 | 1127.83 | 0.00 | 115904.00 | 0.00 |
| **DACSS-1** | 2446.00 | 33.01 | 2847.31 | 155.54 | 20168.44 | 6446.40 | 93073.50 | 5131.84 |
| **DACSS-2** | 2416.39 | 34.69 | 3102.02 | 861.94 | 21725.23 | 5218.53 | 88484.18 | 6099.96 |
| **DACSS-3** | 2462.06 | 14.27 | 2663.92 | 135.37 | 17680.17 | 4584.93 | 86179.51 | 6387.78 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 724.50 | 0.00 | 771.00 | 0.00 | 39913.00 | 0.00 | 39023.80 | 0.00 |
| **DACSS-1** | 2458.07 | 18.96 | 2683.57 | 20.23 | 41461.77 | 56.77 | 81073.08 | 7159.98 |
| **DACSS-2** | 2472.65 | 11.55 | 2649.11 | 46.62 | 41475.79 | 13.27 | 74201.62 | 12690.19 |
| **DACSS-3** | 2396.89 | 22.17 | 2542.72 | 91.39 | 41444.53 | 24.24 | 72839.56 | 5092.08 |

**Table 126. Dynamic DACSS Solution Cost Results - File 4 (15 Nodes)**

| Solution Cost Results - 15 Node DACSS (File 4) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 42836.00 | 0.00 | 42836.00 | 0.00 | 42836.00 | 0.00 |
| **DACSS-1** | 47446.90 | 4784.36 | 44486.52 | 66.56 | 44704.72 | 913.51 | 44318.87 | 77.64 |
| **DACSS-2** | 44402.14 | 33.95 | 44425.48 | 31.06 | 44433.48 | 48.87 | 44468.92 | 73.53 |
| **DACSS-3** | 44291.88 | 27.80 | 44278.72 | 45.48 | 44339.44 | 51.27 | 44367.52 | 86.39 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 52838.50 | 0.00 | 51858.80 | 0.00 | 51833.00 | 0.00 |
| **DACSS-1** | 44465.97 | 125.43 | 54408.59 | 51.54 | 54297.51 | 2066.83 | 54515.34 | 2051.13 |
| **DACSS-2** | 44429.78 | 37.60 | 54381.09 | 110.36 | 53322.89 | 58.24 | 54303.47 | 1472.12 |
| **DACSS-3** | 44369.87 | 44.34 | 54273.18 | 67.60 | 53328.72 | 87.13 | 54612.72 | 1544.23 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 42869.30 | 0.00 | 987.53 | 0.00 | 1012.00 | 0.00 |
| **DACSS-1** | 44439.51 | 42.39 | 44456.16 | 67.42 | 16370.06 | 7728.89 | 2735.34 | 112.35 |
| **DACSS-2** | 44412.92 | 69.97 | 44439.41 | 63.34 | 19161.60 | 6494.38 | 2703.33 | 71.76 |
| **DACSS-3** | 44359.57 | 59.46 | 44471.13 | 34.69 | 11034.38 | 5032.37 | 2670.17 | 75.35 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 85712.00 | 0.00 | 76762.40 | 0.00 | 95755.50 | 0.00 |
| **DACSS-1** | 44525.22 | 31.18 | 87725.77 | 974.53 | 78084.26 | 17.56 | 116053.50 | 2237.42 |
| **DACSS-2** | 44490.20 | 79.30 | 88710.80 | 1671.46 | 78091.30 | 23.49 | 117025.40 | 1545.98 |
| **DACSS-3** | 44426.44 | 121.25 | 87652.43 | 994.14 | 78076.16 | 27.91 | 114487.40 | 2117.63 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 904.00 | 0.00 | 861.33 | 0.00 | 77844.50 | 0.00 |
| **DACSS-1** | 44439.66 | 56.21 | 3383.63 | 1277.91 | 2802.12 | 70.45 | 61615.61 | 3782.54 |
| **DACSS-2** | 44377.40 | 109.86 | 3317.91 | 1349.00 | 3381.26 | 1241.53 | 64816.16 | 5003.94 |
| **DACSS-3** | 44374.08 | 53.69 | 2680.22 | 55.71 | 2780.98 | 39.71 | 58833.05 | 7523.89 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 42836.00 | 0.00 | 817.80 | 0.00 | 46979.70 | 0.00 | 75918.00 | 0.00 |
| **DACSS-1** | 44502.91 | 74.24 | 2497.81 | 22.81 | 50453.68 | 2179.97 | 206467.70 | 8052.63 |
| **DACSS-2** | 44512.51 | 114.73 | 2484.04 | 42.85 | 51304.78 | 2086.30 | 211058.10 | 1213.06 |
| **DACSS-3** | 44381.34 | 86.64 | 2371.22 | 89.62 | 49313.73 | 2293.61 | 203038.60 | 6412.95 |

## Table 127. Dynamic DACSS Solution Cost Results - File 5 (15 Nodes)

| Solution Cost Results - 15 Node DACSS (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 747.00 | 0.00 | 747.00 | 0.00 | 747.00 | 0.00 |
| **DACSS-1** | 2403.59 | 28.79 | 2348.06 | 30.95 | 2332.93 | 50.70 | 2393.49 | 33.98 |
| **DACSS-2** | 2372.09 | 26.87 | 2406.18 | 13.25 | 2373.81 | 32.12 | 2427.92 | 45.22 |
| **DACSS-3** | 2320.04 | 35.86 | 2296.67 | 40.20 | 2318.78 | 66.92 | 2357.11 | 55.78 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 776.60 | 0.00 | 790.00 | 0.00 | 799.25 | 0.00 |
| **DACSS-1** | 2426.00 | 21.09 | 2543.64 | 27.54 | 2391.96 | 40.60 | 2488.41 | 18.84 |
| **DACSS-2** | 2389.92 | 34.49 | 2560.64 | 71.56 | 2345.91 | 57.03 | 2470.42 | 35.02 |
| **DACSS-3** | 2354.28 | 42.83 | 2462.89 | 68.06 | 2320.98 | 33.46 | 2423.39 | 49.78 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 775.33 | 0.00 | 970.40 | 0.00 | 43834.20 | 0.00 |
| **DACSS-1** | 2463.36 | 63.55 | 3066.43 | 972.26 | 2817.81 | 36.34 | 47369.74 | 2336.79 |
| **DACSS-2** | 2471.81 | 56.10 | 2763.99 | 91.20 | 2831.61 | 39.93 | 48391.52 | 2318.01 |
| **DACSS-3** | 2428.01 | 52.08 | 2657.43 | 134.47 | 2793.59 | 96.15 | 46856.54 | 1807.75 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 749.92 | 0.00 | 764.00 | 0.00 | 794.40 | 0.00 |
| **DACSS-1** | 2440.68 | 29.43 | 2594.94 | 47.05 | 8062.12 | 6147.64 | 8936.93 | 6043.33 |
| **DACSS-2** | 2493.62 | 21.04 | 2595.67 | 50.61 | 6978.53 | 4493.82 | 7116.35 | 7504.62 |
| **DACSS-3** | 2424.91 | 55.29 | 2448.29 | 70.36 | 6342.14 | 3043.82 | 4722.28 | 2943.77 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 768.40 | 0.00 | 832.00 | 0.00 | 1009.88 | 0.00 |
| **DACSS-1** | 2475.14 | 36.61 | 2313.91 | 225.92 | 2590.90 | 31.66 | 49738.14 | 6269.46 |
| **DACSS-2** | 2491.16 | 48.22 | 2360.63 | 82.20 | 2610.35 | 29.35 | 54392.12 | 2978.52 |
| **DACSS-3** | 2348.54 | 35.53 | 2363.13 | 42.18 | 2532.84 | 73.70 | 54555.37 | 7371.59 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 747.00 | 0.00 | 765.50 | 0.00 | 49890.00 | 0.00 | 126821.00 | 0.00 |
| **DACSS-1** | 2517.75 | 47.20 | 4053.86 | 2740.67 | 53331.99 | 2202.48 | 156138.80 | 5209.32 |
| **DACSS-2** | 2547.54 | 36.67 | 3757.50 | 2094.88 | 51747.32 | 44.69 | 158567.90 | 2882.73 |
| **DACSS-3** | 2465.99 | 52.08 | 2613.85 | 70.45 | 52286.19 | 1305.61 | 156862.00 | 4879.85 |

Table 128. Dynamic DACSS Solution Cost Results - File 6 (15 Nodes)

| Solution Cost Results - 15 Node DACSS (File 6) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 811.25 | 0.00 | 811.25 | 0.00 | 811.25 | 0.00 |
| **DACSS-1** | 2481.45 | 56.62 | 2410.66 | 62.51 | 2460.75 | 34.88 | 2396.85 | 55.21 |
| **DACSS-2** | 2536.88 | 32.48 | 2519.75 | 43.63 | 2500.00 | 58.19 | 2489.46 | 30.85 |
| **DACSS-3** | 2433.02 | 84.73 | 2454.54 | 77.68 | 2412.80 | 44.37 | 2437.38 | 122.19 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 879.00 | 0.00 | 1022.02 | 0.00 | 1107.87 | 0.00 |
| **DACSS-1** | 2499.31 | 63.92 | 3503.43 | 1570.29 | 2682.56 | 58.44 | 9760.02 | 1442.42 |
| **DACSS-2** | 2549.04 | 39.29 | 3415.89 | 1432.05 | 2689.47 | 63.08 | 8874.04 | 3918.15 |
| **DACSS-3** | 2520.23 | 21.58 | 2639.78 | 102.05 | 2736.02 | 49.97 | 5383.57 | 1394.40 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 853.15 | 0.00 | 55158.60 | 0.00 | 50920.20 | 0.00 |
| **DACSS-1** | 2538.39 | 50.08 | 2801.41 | 138.36 | 7411.58 | 539.80 | 16369.16 | 5525.75 |
| **DACSS-2** | 2497.56 | 100.39 | 2748.50 | 82.01 | 7136.10 | 3463.92 | 13549.21 | 5583.03 |
| **DACSS-3** | 2490.14 | 72.05 | 2716.94 | 73.16 | 5078.92 | 2894.87 | 9832.24 | 2871.68 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 772.00 | 0.00 | 13766.80 | 0.00 | 13790.00 | 0.00 |
| **DACSS-1** | 2565.68 | 45.91 | 2652.59 | 78.85 | 5143.05 | 2621.90 | 11181.78 | 5374.33 |
| **DACSS-2** | 2518.55 | 53.51 | 2611.58 | 67.07 | 4136.13 | 2350.51 | 19919.50 | 2404.16 |
| **DACSS-3** | 2532.01 | 63.00 | 2522.27 | 61.59 | 3638.26 | 2030.46 | 9376.85 | 3163.48 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 898.00 | 0.00 | 888.50 | 0.00 | 2960.33 | 0.00 |
| **DACSS-1** | 2603.40 | 41.03 | 2909.31 | 49.77 | 2842.32 | 91.73 | 2797.49 | 74.60 |
| **DACSS-2** | 2618.61 | 57.44 | 3782.72 | 1693.09 | 2832.41 | 136.02 | 2829.50 | 83.54 |
| **DACSS-3** | 2605.42 | 43.35 | 3418.64 | 1324.13 | 2815.74 | 143.94 | 2776.89 | 96.79 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 811.25 | 0.00 | 833.00 | 0.00 | 908.35 | 0.00 | 133843.00 | 0.00 |
| **DACSS-1** | 2635.56 | 42.44 | 2627.73 | 153.71 | 2740.35 | 65.54 | 144122.20 | 2344.72 |
| **DACSS-2** | 2681.98 | 17.76 | 3188.95 | 1280.04 | 2767.84 | 95.81 | 143007.60 | 3.10 |
| **DACSS-3** | 2591.58 | 59.60 | 3083.02 | 1222.53 | 2633.85 | 85.50 | 148210.40 | 7012.14 |

## Table 129. Dynamic DACSS Solution Cost Results - File 7 (15 Nodes)

| Solution Cost Results - 15 Node DACSS (File 7) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 829.80 | 0.00 | 829.80 | 0.00 | 829.80 | 0.00 |
| **DACSS-1** | 2372.41 | 82.40 | 2407.56 | 42.40 | 2410.62 | 64.05 | 2436.07 | 51.59 |
| **DACSS-2** | 2366.23 | 41.82 | 2444.15 | 85.21 | 2390.33 | 117.54 | 2450.20 | 38.17 |
| **DACSS-3** | 2442.63 | 51.84 | 2432.69 | 43.75 | 2413.94 | 64.95 | 2345.62 | 26.16 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 856.00 | 0.00 | 834.40 | 0.00 | 818.90 | 0.00 |
| **DACSS-1** | 2480.53 | 30.29 | 2430.51 | 59.88 | 2439.13 | 111.48 | 2925.06 | 972.27 |
| **DACSS-2** | 2489.02 | 68.15 | 2427.43 | 86.81 | 2466.88 | 159.24 | 2557.04 | 113.95 |
| **DACSS-3** | 2451.81 | 46.73 | 2337.76 | 72.64 | 2482.58 | 25.51 | 2456.85 | 82.47 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 838.00 | 0.00 | 1168.85 | 0.00 | 1084.87 | 0.00 |
| **DACSS-1** | 2492.23 | 70.11 | 2649.43 | 174.81 | 7132.66 | 2684.17 | 3063.54 | 193.91 |
| **DACSS-2** | 2378.39 | 174.67 | 2621.45 | 181.91 | 6404.13 | 2403.27 | 5714.59 | 2461.10 |
| **DACSS-3** | 2483.32 | 99.37 | 2677.10 | 22.44 | 6896.73 | 3157.23 | 3766.71 | 1606.15 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 784.00 | 0.00 | 901.00 | 0.00 | 936.20 | 0.00 |
| **DACSS-1** | 2505.92 | 20.90 | 2603.16 | 77.46 | 23219.62 | 7727.55 | 11422.69 | 6839.86 |
| **DACSS-2** | 2414.97 | 169.47 | 2570.07 | 26.97 | 36898.87 | 9264.66 | 12090.64 | 6151.95 |
| **DACSS-3** | 2465.28 | 57.63 | 2553.24 | 47.92 | 14494.66 | 10542.07 | 8011.23 | 5603.51 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 864.67 | 0.00 | 895.00 | 0.00 | 52112.20 | 0.00 |
| **DACSS-1** | 2329.52 | 249.47 | 2779.74 | 60.65 | 2779.14 | 81.94 | 9935.13 | 3167.13 |
| **DACSS-2** | 2498.60 | 65.43 | 2778.93 | 86.75 | 3569.96 | 1657.47 | 7581.05 | 3566.05 |
| **DACSS-3** | 2449.56 | 126.02 | 2693.22 | 44.45 | 2642.78 | 145.14 | 6683.17 | 2379.30 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 829.80 | 0.00 | 900.80 | 0.00 | 807.93 | 0.00 | 985.75 | 0.00 |
| **DACSS-1** | 2431.44 | 102.78 | 6251.04 | 570.08 | 2735.86 | 106.70 | 3632.14 | 1328.56 |
| **DACSS-2** | 2457.74 | 132.04 | 7865.34 | 3018.70 | 2727.44 | 24.90 | 4802.05 | 3007.55 |
| **DACSS-3** | 2443.73 | 89.98 | 4624.27 | 2342.05 | 2648.63 | 70.63 | 3008.11 | 19.23 |

**Table 130. Dynamic DACSS Solution Cost Results - File 8 (15 Nodes)**

| Solution Cost Results - 15 Node DACSS (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 788.25 | 0.00 | 788.25 | 0.00 | 788.25 | 0.00 |
| **DACSS-1** | 2605.05 | 39.18 | 2593.83 | 68.86 | 2536.91 | 56.97 | 2547.96 | 48.18 |
| **DACSS-2** | 2621.66 | 32.39 | 2578.54 | 31.95 | 2566.81 | 25.09 | 2619.34 | 55.51 |
| **DACSS-3** | 2583.62 | 58.13 | 2484.35 | 80.46 | 2532.45 | 69.86 | 2395.52 | 140.93 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 840.75 | 0.00 | 933.70 | 0.00 | 925.17 | 0.00 |
| **DACSS-1** | 2682.56 | 17.98 | 2656.17 | 56.08 | 2850.26 | 74.03 | 2875.91 | 61.38 |
| **DACSS-2** | 2633.42 | 90.37 | 2650.46 | 38.45 | 2943.99 | 91.30 | 2849.89 | 66.80 |
| **DACSS-3** | 2556.49 | 41.82 | 2606.24 | 52.80 | 2766.60 | 31.41 | 2690.11 | 91.94 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 825.00 | 0.00 | 48864.50 | 0.00 | 77794.00 | 0.00 |
| **DACSS-1** | 2572.40 | 64.63 | 5308.81 | 2353.16 | 50556.52 | 78.21 | 53535.44 | 3520.27 |
| **DACSS-2** | 2572.95 | 21.47 | 6066.91 | 1294.57 | 50640.88 | 50.89 | 54413.01 | 2053.76 |
| **DACSS-3** | 2294.98 | 459.97 | 3073.74 | 1355.30 | 50571.63 | 53.00 | 51428.55 | 120.09 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 836.25 | 0.00 | 44774.60 | 0.00 | 45835.40 | 0.00 |
| **DACSS-1** | 2428.21 | 433.67 | 2601.55 | 27.74 | 46433.52 | 52.58 | 49337.72 | 2346.55 |
| **DACSS-2** | 2620.59 | 50.60 | 2577.97 | 49.71 | 46502.16 | 71.00 | 48320.50 | 1538.95 |
| **DACSS-3** | 2493.42 | 238.18 | 2567.34 | 52.30 | 46692.56 | 970.85 | 47599.78 | 65.22 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 764.45 | 0.00 | 833.20 | 0.00 | 58801.20 | 0.00 |
| **DACSS-1** | 2656.62 | 70.96 | 2484.92 | 43.24 | 4733.16 | 2634.69 | 90239.99 | 26.25 |
| **DACSS-2** | 2647.14 | 46.59 | 2469.23 | 43.31 | 6364.22 | 1976.48 | 90260.07 | 22.85 |
| **DACSS-3** | 2597.26 | 135.53 | 2433.67 | 45.50 | 7631.66 | 3891.52 | 90209.89 | 33.96 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 788.25 | 0.00 | 925.20 | 0.00 | 15849.10 | 0.00 | 1928.50 | 0.00 |
| **DACSS-1** | 2627.81 | 82.22 | 3305.08 | 827.24 | 8745.17 | 1999.55 | 108778.86 | 8989.31 |
| **DACSS-2** | 2625.76 | 54.42 | 2898.84 | 43.58 | 7533.17 | 3063.81 | 106652.34 | 16053.90 |
| **DACSS-3** | 2595.82 | 68.94 | 2832.69 | 115.60 | 6167.46 | 1274.17 | 103932.24 | 10147.10 |

**Table 131. Dynamic DACSS Solution Cost Results - File 9 (15 Nodes)**

| Solution Cost Results - 15 Node DACSS (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 730.60 | 0.00 | 730.60 | 0.00 | 730.60 | 0.00 |
| **DACSS-1** | 2194.22 | 15.84 | 2192.19 | 5.91 | 2196.19 | 24.39 | 2140.77 | 79.01 |
| **DACSS-2** | 2183.39 | 25.44 | 2183.21 | 26.66 | 2203.00 | 34.71 | 2211.41 | 10.60 |
| **DACSS-3** | 2184.75 | 29.75 | 2183.76 | 52.44 | 2136.65 | 24.80 | 2169.09 | 19.09 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 816.00 | 0.00 | 865.25 | 0.00 | 856.75 | 0.00 |
| **DACSS-1** | 2258.08 | 34.34 | 2382.86 | 66.39 | 5985.19 | 6098.43 | 3395.42 | 1260.20 |
| **DACSS-2** | 2229.97 | 43.89 | 2378.96 | 31.52 | 9145.72 | 5410.46 | 2702.38 | 74.01 |
| **DACSS-3** | 2168.08 | 38.45 | 2376.23 | 34.16 | 3746.20 | 1459.66 | 3134.16 | 1611.32 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 755.67 | 0.00 | 741.00 | 0.00 | 2845.90 | 0.00 |
| **DACSS-1** | 2187.67 | 39.35 | 2292.96 | 26.93 | 2385.48 | 21.04 | 2566.03 | 135.26 |
| **DACSS-2** | 2184.85 | 28.48 | 2320.52 | 9.84 | 2383.98 | 30.22 | 2531.18 | 152.95 |
| **DACSS-3** | 2179.28 | 29.04 | 2305.08 | 31.57 | 2322.90 | 24.04 | 2610.99 | 55.04 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 748.92 | 0.00 | 806.20 | 0.00 | 868.00 | 0.00 |
| **DACSS-1** | 2307.85 | 44.56 | 2416.10 | 69.58 | 6017.67 | 3874.68 | 10475.11 | 2055.33 |
| **DACSS-2** | 2320.06 | 52.98 | 2534.67 | 54.56 | 7103.43 | 1625.21 | 15069.19 | 6655.90 |
| **DACSS-3** | 2280.26 | 46.47 | 2462.77 | 66.58 | 4205.79 | 2073.12 | 10408.65 | 3772.90 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 818.25 | 0.00 | 816.20 | 0.00 | 24820.10 | 0.00 |
| **DACSS-1** | 2275.86 | 27.93 | 2546.95 | 33.59 | 2367.52 | 13.34 | 17705.05 | 2682.74 |
| **DACSS-2** | 2291.37 | 25.46 | 2577.12 | 24.58 | 2371.90 | 23.09 | 19695.03 | 3886.69 |
| **DACSS-3** | 2253.98 | 47.79 | 2534.41 | 43.18 | 2327.26 | 33.07 | 15834.50 | 2689.87 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 730.60 | 0.00 | 851.00 | 0.00 | 950.30 | 0.00 | 117848.00 | 0.00 |
| **DACSS-1** | 2351.38 | 20.96 | 2647.20 | 47.03 | 15595.05 | 3154.45 | 170481.70 | 13615.21 |
| **DACSS-2** | 2331.00 | 9.53 | 2652.98 | 46.94 | 21802.18 | 3408.11 | 152604.00 | 4861.82 |
| **DACSS-3** | 2216.96 | 115.89 | 2517.78 | 121.64 | 16779.89 | 3881.69 | 156808.20 | 4883.70 |

## Table 132. Dynamic DACSS Solution Cost Results - File 10 (15 Nodes)

| Solution Cost Results - 15 Node DACSS (File 10) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 745.00 | 0.00 | 745.00 | 0.00 | 745.00 | 0.00 |
| **DACSS-1** | 2345.45 | 16.20 | 2325.61 | 19.72 | 2301.32 | 17.39 | 2362.99 | 34.83 |
| **DACSS-2** | 2381.26 | 16.71 | 2378.86 | 7.58 | 2378.51 | 24.70 | 2313.50 | 75.36 |
| **DACSS-3** | 2274.42 | 19.24 | 2294.30 | 34.90 | 2218.38 | 143.19 | 2268.54 | 26.44 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 755.80 | 0.00 | 773.60 | 0.00 | 762.05 | 0.00 |
| **DACSS-1** | 2400.12 | 56.63 | 2359.53 | 123.94 | 2465.47 | 36.08 | 2299.04 | 217.66 |
| **DACSS-2** | 2428.44 | 33.62 | 2473.86 | 38.42 | 2383.86 | 137.98 | 2400.66 | 95.16 |
| **DACSS-3** | 2349.78 | 50.54 | 2395.55 | 50.67 | 2384.75 | 46.48 | 2398.39 | 118.36 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 750.00 | 0.00 | 752.50 | 0.00 | 736.20 | 0.00 |
| **DACSS-1** | 2417.95 | 26.90 | 2377.57 | 65.64 | 2383.71 | 38.46 | 2354.56 | 15.53 |
| **DACSS-2** | 2440.69 | 34.89 | 2368.58 | 117.93 | 2373.11 | 37.54 | 2368.16 | 18.32 |
| **DACSS-3** | 2283.00 | 50.78 | 2399.45 | 29.13 | 2295.91 | 56.19 | 2352.21 | 30.11 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 813.70 | 0.00 | 43833.70 | 0.00 | 31976.20 | 0.00 |
| **DACSS-1** | 2424.48 | 36.28 | 5271.15 | 2351.90 | 21564.58 | 6194.26 | 48044.27 | 3635.19 |
| **DACSS-2** | 2441.56 | 20.69 | 5829.93 | 2791.67 | 24199.97 | 9804.35 | 47167.63 | 6695.48 |
| **DACSS-3** | 2341.55 | 40.46 | 3470.15 | 1913.12 | 19700.99 | 9537.40 | 45830.45 | 3762.04 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 76713.40 | 0.00 | 42806.70 | 0.00 | 20917.00 | 0.00 |
| **DACSS-1** | 2555.40 | 21.61 | 41406.62 | 80.34 | 65244.28 | 11573.60 | 30737.33 | 2716.18 |
| **DACSS-2** | 2499.37 | 38.14 | 41479.66 | 16.79 | 59031.05 | 10297.46 | 31014.19 | 1682.82 |
| **DACSS-3** | 2436.06 | 38.67 | 41365.60 | 59.95 | 61314.15 | 8861.81 | 25507.37 | 5118.11 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | | **Cost/Std Dev** | |
| **MILP** | 745.00 | 0.00 | 886.20 | 0.00 | 83848.90 | 0.00 | 10029.20 | 0.00 |
| **DACSS-1** | 2505.86 | 24.34 | 2879.83 | 119.04 | 84062.01 | 1416.39 | 53031.58 | 3589.56 |
| **DACSS-2** | 2508.58 | 23.78 | 3619.69 | 1788.99 | 68232.72 | 34684.10 | 44362.02 | 20388.23 |
| **DACSS-3** | 2437.56 | 47.57 | 2790.00 | 135.74 | 84206.95 | 1519.50 | 54835.34 | 5869.73 |

## Appendix G: *t*-Tests ACSE/DACSE (10 Nodes)

### Table 133. *t*-Test ACSE/DACSE - File 1 (10 Nodes)

| ***t*-Test & DACSE/ACSE Mean Ratio Results - 10 Node (File 1)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3300 | 1.0477 | 0.1600 | 1.0964 | 1.0000 | 0.9998 | 0.3200 | 0.9540 |
| **2** | 0.1600 | 1.0965 | 0.3200 | 1.0484 | 0.6000 | 1.0001 | 0.5600 | 0.9555 |
| **3** | 0.3200 | 0.9538 | 0.8900 | 1.0150 | 0.8500 | 1.0196 | 0.1200 | 1.1655 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7100 | 0.9999 | 0.2100 | 1.0075 | 0.3300 | 0.9985 | 0.5400 | 0.8618 |
| **2** | 0.3300 | 1.0482 | 0.6300 | 0.9973 | 1.0000 | 1.0002 | 0.3000 | 1.3570 |
| **3** | 0.6900 | 0.9781 | 0.3300 | 0.7886 | 0.3300 | 0.7076 | 0.3500 | 1.2619 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 1.0000 | 0.9998 | 0.8200 | 0.9975 | 0.1700 | 0.9912 | **0.0400** | 0.9998 |
| **2** | 0.3300 | 1.0471 | 0.6300 | 0.9949 | 0.2000 | 1.0120 | 0.2200 | 0.9770 |
| **3** | 0.7900 | 0.9795 | 0.1600 | 0.9512 | 0.7900 | 0.9970 | 0.3000 | 0.9999 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4900 | 1.0002 | 0.3600 | 1.0198 | 0.3100 | 1.0050 | 0.1500 | 1.1168 |
| **2** | 0.0600 | 0.9994 | 0.9000 | 0.9968 | **0.0200** | 1.0005 | **0.0000** | 0.7157 |
| **3** | 0.1800 | 1.0760 | 0.2300 | 1.0207 | 0.3900 | 0.9879 | 0.8800 | 1.0136 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4300 | 1.0003 | 0.7600 | 0.9485 | 0.1800 | 1.0266 | 0.6700 | 1.0000 |
| **2** | 0.2800 | 1.0004 | 0.1200 | 1.2483 | 0.7500 | 0.9920 | 0.1100 | 1.0000 |
| **3** | 0.5000 | 1.0532 | 0.4700 | 0.8622 | 0.8700 | 1.0030 | 0.8800 | 1.0000 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7300 | 1.0001 | 0.3000 | 0.9920 | 0.3300 | 0.9947 | 1.0000 | 1.0000 |
| **2** | 0.3300 | 1.0485 | 0.8400 | 1.0009 | 0.3200 | 1.0054 | 0.8000 | 1.0280 |
| **3** | 0.5700 | 0.9585 | 0.4500 | 1.0051 | 1.0000 | 1.0000 | 0.7400 | 0.9651 |

**Table 134. *t*-Test ACSE/DACSE - File 2 (10 Nodes)**

| *t*-Test & DACSE/ACSE Mean Ratio Results - 10 Node (File 2) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3500 | 1.2189 | 0.7400 | 1.0477 | 0.9400 | 0.9887 | 0.4500 | 1.0991 |
| **2** | 0.9600 | 0.9917 | 0.4200 | 0.8404 | 0.1300 | 0.7373 | 0.7900 | 0.9462 |
| **3** | 0.8200 | 0.9380 | 0.9800 | 1.0045 | 0.5900 | 0.8945 | 0.1900 | 0.8065 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3000 | 0.8327 | 0.2000 | 1.4657 | 0.2900 | 0.8818 | 0.1000 | 0.8655 |
| **2** | 0.6500 | 0.9159 | 0.8000 | 1.0772 | **0.0100** | 1.3621 | 0.9700 | 1.0034 |
| **3** | 0.6900 | 0.9195 | 0.6300 | 0.8804 | 0.5700 | 0.9323 | 0.6900 | 1.0466 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9500 | 0.9892 | 0.7300 | 1.0540 | 0.1400 | 0.8358 | 1.0000 | 1.0000 |
| **2** | 0.4600 | 0.8712 | **0.0300** | 0.6397 | 0.6200 | 0.9190 | 0.1400 | 1.0419 |
| **3** | 0.5000 | 1.1287 | 0.8700 | 1.0397 | 0.6200 | 1.0631 | 0.8500 | 1.0050 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7000 | 0.9359 | 0.0700 | 0.8040 | 0.1700 | 0.7106 | 0.1900 | 0.9877 |
| **2** | 0.7000 | 0.9265 | 0.0700 | 0.7912 | 0.4000 | 0.7392 | 0.4600 | 1.0045 |
| **3** | 0.6300 | 1.1237 | 0.7200 | 1.0442 | 0.8500 | 1.0507 | 0.2500 | 1.0116 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5300 | 0.8887 | **0.0200** | 1.0876 | 1.0000 | 1.0000 | **0.0200** | 0.9998 |
| **2** | 0.6400 | 1.0938 | 0.5000 | 0.9825 | 1.0000 | 1.0000 | 0.9300 | 1.0000 |
| **3** | 0.6000 | 1.1067 | 0.9200 | 1.0045 | 1.0000 | 1.0000 | 0.2600 | 0.9999 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5000 | 1.1494 | 1.0000 | 0.9999 | 0.2500 | 1.0330 | 1.0000 | 1.0000 |
| **2** | 0.7400 | 0.9377 | **0.0400** | 1.3052 | 0.1300 | 1.0972 | 1.0000 | 1.0000 |
| **3** | 0.8500 | 0.9563 | 1.0000 | 1.0000 | 0.4100 | 1.0285 | 1.0000 | 1.0000 |

**Table 135. *t*-Test ACSE/DACSE - File 3 (10 Nodes)**

| *t*-Test & DACSE/ACSE Mean Ratio Results - 10 Node (File 3) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7400 | 1.1417 | 0.6200 | 0.9979 | 0.3100 | 1.1677 | **0.0100** | 1.0075 |
| **2** | 1.0000 | 1.0005 | 0.1100 | 0.5331 | 0.2300 | 1.7419 | 0.0900 | 2.3090 |
| **3** | 0.9900 | 1.0040 | 0.2300 | 1.0096 | 0.9700 | 1.0067 | 0.5500 | 0.7982 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5700 | 0.7551 | 0.3500 | 0.8150 | 0.3000 | 0.8611 | 0.5500 | 0.9994 |
| **2** | 0.3100 | 1.6491 | 0.7400 | 0.9200 | 0.7300 | 0.9972 | **0.0400** | 1.0059 |
| **3** | 0.9100 | 0.9524 | 0.6800 | 1.0056 | 0.3100 | 1.2872 | 0.7900 | 1.0011 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4500 | 1.3857 | 0.7600 | 1.0012 | 0.3000 | 1.0196 | 0.3100 | 1.0001 |
| **2** | 0.2900 | 1.5839 | 0.0900 | 1.0066 | **0.0400** | 1.0288 | 0.0600 | 0.9999 |
| **3** | 0.2700 | 0.6537 | 0.3100 | 1.1319 | 0.7200 | 0.9934 | 0.8400 | 1.0000 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4400 | 1.3431 | 0.1400 | 1.0050 | 0.8900 | 1.0335 | 0.3300 | 1.0000 |
| **2** | 0.0700 | 2.2785 | 0.3500 | 1.0068 | 0.7100 | 1.0652 | 0.3300 | 1.0000 |
| **3** | 0.9100 | 1.0475 | 0.5900 | 0.9977 | 0.8400 | 0.9559 | 0.3300 | 1.0000 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5400 | 1.2803 | 0.0500 | 1.4643 | 0.4700 | 1.1264 | 1.0000 | 1.0000 |
| **2** | 0.1500 | 1.8685 | 0.3000 | 0.7971 | 0.2400 | 0.7846 | 1.0000 | 1.0000 |
| **3** | 0.2100 | 0.6148 | 0.1100 | 1.4379 | 0.2400 | 0.8008 | 1.0000 | 1.0000 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2300 | 1.7057 | 0.5000 | 0.6878 | 0.8600 | 1.0000 | **0.0400** | 0.9363 |
| **2** | 0.1300 | 1.9390 | 0.4000 | 1.3157 | 0.6900 | 1.0000 | 0.3100 | 0.9850 |
| **3** | 0.2800 | 1.5403 | 0.3800 | 0.6077 | 0.6400 | 1.0001 | 0.1200 | 0.9502 |

Table 136. *t*-Test ACSE/DACSE - File 4 (10 Nodes)

| *t*-Test & DACSE/ACSE Mean Ratio Results - 10 Node (File 4) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6700 | 1.0670 | 0.9800 | 0.9962 | 0.2300 | 1.2223 | 0.0700 | 0.7397 |
| **2** | 0.3400 | 0.8877 | 0.9800 | 1.0039 | 0.4100 | 1.1372 | 0.4400 | 1.1031 |
| **3** | 0.8000 | 1.0410 | 0.4700 | 0.8659 | 0.6600 | 0.9158 | 0.8500 | 1.0310 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5500 | 1.0919 | 0.4200 | 1.1210 | 0.5900 | 0.8996 | 0.5100 | 0.9979 |
| **2** | 0.7500 | 0.9583 | 0.4400 | 1.0901 | 0.4800 | 1.2060 | 0.5700 | 0.9974 |
| **3** | 0.6100 | 1.0844 | 0.8100 | 0.9544 | 0.9600 | 0.9842 | 0.5000 | 0.9959 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3400 | 1.1391 | 0.2200 | 0.9131 | **0.0100** | 0.9991 | 0.3700 | 1.0393 |
| **2** | 0.8300 | 1.0359 | 0.9400 | 0.9946 | 0.6000 | 0.9994 | 0.9400 | 0.9974 |
| **3** | 0.4900 | 1.1126 | 0.5800 | 0.9558 | 0.0800 | 0.9990 | 0.2900 | 1.0447 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9200 | 1.0130 | 0.2800 | 1.0690 | 0.4600 | 1.0777 | 0.8600 | 0.9979 |
| **2** | 0.6600 | 1.0702 | 0.1400 | 0.8645 | 0.3300 | 0.8776 | 0.1300 | 1.0159 |
| **3** | 0.6500 | 1.0713 | 0.8300 | 1.0108 | 0.6600 | 0.9612 | 0.9700 | 1.0004 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3000 | 1.1242 | 0.5400 | 1.0336 | 0.2900 | 1.0180 | 0.3300 | 1.0005 |
| **2** | 0.1900 | 1.2241 | 0.5100 | 0.9683 | 0.3300 | 0.9972 | 0.1300 | 1.0000 |
| **3** | 0.1500 | 1.2130 | 0.3700 | 0.9555 | 0.9100 | 1.0017 | 0.3200 | 1.0005 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8300 | 0.9718 | 0.0800 | 1.0108 | **0.0100** | 1.0608 | 0.8000 | 1.0019 |
| **2** | 0.3300 | 0.8882 | 0.5500 | 0.9981 | 0.9100 | 0.9962 | 0.2400 | 1.0044 |
| **3** | 0.1400 | 1.2334 | 0.2600 | 1.0061 | 0.3900 | 1.0261 | 0.1300 | 1.0158 |

## Table 137. *t*-Test ACSE/DACSE - File 5 (10 Nodes)

| *t*-Test & DACSE/ACSE Mean Ratio Results - 10 Node (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1000 | 1.0120 | 0.1500 | 1.0034 | 0.0700 | 1.0053 | 0.1100 | 1.0026 |
| **2** | 0.1600 | 1.0091 | 0.6800 | 1.0026 | 0.3800 | 1.0060 | 0.8300 | 1.0014 |
| **3** | 0.6800 | 1.0030 | **0.0300** | 1.0068 | **0.0000** | 1.0106 | **0.0200** | 1.0057 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1600 | 1.0095 | 0.3100 | 1.0375 | 0.2000 | 1.0048 | 0.9600 | 0.9996 |
| **2** | 0.3900 | 1.0062 | 0.2300 | 1.0850 | 0.5500 | 1.0079 | 0.1300 | 0.9781 |
| **3** | 0.0500 | 1.0139 | **0.0400** | 0.8868 | 0.3800 | 1.0024 | 0.0500 | 0.9833 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0200** | 1.0081 | 0.3200 | 1.0220 | 0.4400 | 1.0986 | **0.0200** | 0.8441 |
| **2** | **0.0000** | 1.0105 | 0.9900 | 1.0012 | 0.7600 | 0.9767 | **0.0400** | 0.8353 |
| **3** | 0.4000 | 1.0029 | 0.2600 | 1.0004 | 0.6100 | 1.0617 | 0.3600 | 1.0822 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2800 | 1.0056 | 0.8200 | 0.9890 | 1.0000 | 1.0000 | 0.1600 | 0.9999 |
| **2** | 0.6200 | 1.0025 | 0.6100 | 0.9723 | 1.0000 | 1.0000 | 0.3300 | 0.9999 |
| **3** | 0.2200 | 1.0065 | **0.0400** | 1.1361 | 0.3300 | 1.0000 | 0.3300 | 0.9999 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2700 | 1.0098 | 0.7200 | 1.0008 | 0.1600 | 1.0002 | 1.0000 | 1.0000 |
| **2** | 0.7800 | 0.9974 | 0.8100 | 1.0008 | 0.7700 | 0.9970 | 1.0000 | 1.0000 |
| **3** | 0.6000 | 0.9953 | 0.9800 | 1.0001 | 0.3300 | 0.9970 | 1.0000 | 1.0000 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0600 | 0.9861 | 0.4000 | 1.0000 | 0.7100 | 1.0000 | 1.0000 | 1.0000 |
| **2** | **0.0200** | 1.0198 | 0.1300 | 1.0003 | 0.3400 | 1.0001 | 0.3300 | 1.0000 |
| **3** | 0.5200 | 1.0054 | 0.9600 | 1.0000 | 0.9500 | 1.0000 | 1.0000 | 1.0000 |

## Table 138. *t*-Test ACSE/DACSE - File 6 (10 Nodes)

| *t*-Test & DACSE/ACSE Mean Ratio Results - 10 Node (File 6) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | |
| 1 | 0.0600 | 1.5471 | 0.3300 | 1.2076 | 0.8800 | 0.9659 | 0.2500 | 0.7496 |
| 2 | 0.9900 | 0.9985 | 0.2200 | 1.3031 | 0.2800 | 1.2391 | 0.1800 | 1.2833 |
| 3 | 0.7100 | 1.0777 | 0.9700 | 0.9901 | 0.7200 | 1.0886 | 0.9400 | 1.0183 |
| 10% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | |
| 1 | 0.2900 | 0.7939 | **0.0200** | 0.5470 | 0.7800 | 1.0565 | 0.0500 | 1.3018 |
| 2 | 0.1200 | 1.4579 | 0.7000 | 0.9028 | 0.7900 | 1.0698 | 0.2100 | 1.2188 |
| 3 | 0.4600 | 0.8541 | 0.9900 | 1.0017 | 0.3400 | 0.7838 | 0.9800 | 1.0052 |
| 20% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | |
| 1 | 0.8800 | 0.9683 | 0.1800 | 0.9988 | **0.0300** | 1.0042 | 0.6800 | 1.0000 |
| 2 | 0.0800 | 0.6714 | 0.1400 | 1.5009 | 0.6300 | 1.0019 | 0.0700 | 1.0234 |
| 3 | 0.4700 | 1.1766 | 0.6600 | 0.8880 | 0.6500 | 1.0016 | 0.9700 | 1.0000 |
| 30% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | |
| 1 | 0.2400 | 1.2863 | 0.9100 | 0.9860 | 0.4400 | 1.1423 | **0.0000** | 1.7394 |
| 2 | **0.0000** | 2.0075 | 0.5500 | 1.0332 | 0.2200 | 0.8842 | 0.2400 | 1.3184 |
| 3 | 0.7300 | 1.0843 | 0.2000 | 1.1818 | 0.4400 | 1.1660 | **0.0000** | 1.9300 |
| 40% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | |
| 1 | 0.2800 | 1.2433 | 0.9500 | 0.9994 | 0.1800 | 0.9500 | 0.8200 | 1.0000 |
| 2 | 0.4100 | 1.2052 | 0.4800 | 0.9229 | 0.0700 | 0.9633 | 0.3000 | 1.0000 |
| 3 | 0.9600 | 1.0107 | 0.7200 | 1.0030 | 0.6100 | 1.0316 | 0.3200 | 1.0000 |
| 50% | Iterations 1:25 | | Iterations 26:50 | | Iterations 51:75 | | Iterations 76:100 | |
| METHOD | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | | p-Value/Ratio | |
| 1 | 0.4300 | 0.8668 | 0.8000 | 1.0007 | 0.2800 | 1.1168 | 1.0000 | 1.0000 |
| 2 | 0.4900 | 1.1578 | 0.2900 | 1.0032 | 0.5400 | 0.9597 | 0.2000 | 1.0002 |
| 3 | 0.3900 | 1.2176 | **0.0100** | 1.0131 | 0.6600 | 0.9517 | 1.0000 | 1.0000 |

**Table 139. *t*-Test ACSE/DACSE - File 7 (10 Nodes)**

| *t*-Test & DACSE/ACSE Mean Ratio Results - 10 Node (File 7) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4800 | 1.1774 | 0.3300 | 1.2524 | 0.5200 | 0.8576 | 0.4000 | 1.2231 |
| **2** | 0.6800 | 1.0910 | 0.1500 | 0.6868 | 0.2400 | 0.7797 | **0.0000** | 0.5417 |
| **3** | 0.3700 | 0.7619 | 0.2600 | 0.7481 | 0.9600 | 1.0131 | 0.0900 | 0.6182 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6900 | 0.9274 | 0.2000 | 0.7412 | 0.8300 | 0.9878 | 1.0000 | 0.9998 |
| **2** | 0.4600 | 0.8426 | 0.2900 | 0.7793 | 0.9000 | 1.0065 | 1.0000 | 1.0004 |
| **3** | 0.9900 | 0.9971 | 1.0000 | 0.9991 | **0.0200** | 1.2678 | 0.8000 | 0.9610 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4600 | 1.1292 | 0.1100 | 1.0003 | **0.0400** | 1.0004 | 0.7900 | 1.0000 |
| **2** | 0.4400 | 1.2207 | 0.2600 | 1.0007 | **0.0100** | 1.0007 | **0.0200** | 1.0004 |
| **3** | 0.3700 | 0.7842 | 0.5800 | 1.0001 | 0.2600 | 0.9998 | 0.6900 | 1.0001 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6000 | 1.0991 | 0.5100 | 0.9974 | 0.6900 | 1.0544 | 1.0000 | 1.0000 |
| **2** | 0.3500 | 1.2077 | 0.7800 | 1.0012 | 0.8900 | 1.0306 | 0.3300 | 0.9734 |
| **3** | 0.5800 | 1.1428 | 0.7500 | 0.9985 | 0.5900 | 1.0665 | 1.0000 | 1.0000 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5100 | 0.8617 | 0.1900 | 1.0078 | 0.4500 | 0.9212 | **0.0300** | 1.0647 |
| **2** | 0.4800 | 1.1483 | 0.3200 | 1.0066 | 0.1300 | 0.8438 | 0.0900 | 1.0846 |
| **3** | 0.4400 | 1.2893 | 0.2800 | 0.9933 | **0.0100** | 0.7598 | 0.4800 | 0.9794 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4600 | 1.1973 | 0.3300 | 0.9436 | 0.1200 | 0.9998 | 1.0000 | 1.0000 |
| **2** | 0.9600 | 0.9879 | 0.6700 | 0.9489 | 0.3400 | 1.0000 | 1.0000 | 1.0000 |
| **3** | **0.0300** | 0.6216 | 0.4700 | 0.9394 | 0.1700 | 0.9998 | 1.0000 | 1.0000 |

**Table 140. *t*-Test ACSE/DACSE - File 8 (10 Nodes)**

| *t*-Test & DACSE/ACSE Mean Ratio Results - 10 Node (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2400 | 0.7211 | 0.7400 | 0.9230 | 0.7600 | 0.9317 | 0.5900 | 0.8904 |
| **2** | 0.4100 | 0.8387 | 0.3300 | 1.2848 | 0.0500 | 1.7136 | 0.2100 | 1.3871 |
| **3** | 0.3600 | 0.7849 | 0.2400 | 1.3069 | 0.7900 | 0.9410 | 0.9900 | 1.0028 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5000 | 1.1683 | 0.8200 | 0.9273 | 0.0900 | 1.0005 | 0.4700 | 1.0001 |
| **2** | 0.1700 | 0.7381 | 0.2400 | 0.7124 | 0.5800 | 0.9999 | **0.0300** | 1.0008 |
| **3** | 0.2800 | 0.7774 | 0.8800 | 0.9444 | 0.8600 | 1.0000 | 0.9200 | #VALUE! |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9900 | 0.9954 | 0.1200 | 2.9815 | 0.5400 | 0.9998 | 0.3200 | 0.9823 |
| **2** | 0.8100 | 0.9347 | 0.1700 | 1.5045 | 0.2500 | 0.9997 | 0.5400 | 1.0115 |
| **3** | 0.0900 | 0.6587 | 1.0000 | 0.9998 | 0.6000 | 1.0002 | 0.1300 | 0.9713 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3000 | 0.7814 | 0.5600 | 1.2773 | 0.1900 | 1.0416 | 0.3300 | 0.9980 |
| **2** | 0.8200 | 0.9417 | 0.8600 | 0.9552 | 0.0900 | 1.1211 | 0.8000 | 1.0021 |
| **3** | 0.8800 | 1.0396 | 0.7400 | 1.1438 | 0.1500 | 1.0752 | 0.4700 | 0.9960 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7700 | 0.9360 | 0.5800 | 1.0076 | 0.5600 | 0.8853 | 0.1400 | 1.1503 |
| **2** | 0.1700 | 0.6655 | 0.4100 | 0.8686 | 0.2100 | 1.2603 | 0.2300 | 0.5028 |
| **3** | 0.6600 | 0.8861 | **0.0100** | 1.0289 | 0.3300 | 1.1480 | **0.0400** | 1.3040 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6600 | 1.1033 | 0.5700 | 1.0672 | 0.2000 | 1.0213 | 0.3800 | 0.9961 |
| **2** | 0.7700 | 1.0607 | 0.3600 | 1.0932 | 1.0000 | 1.0000 | 0.1600 | 0.9974 |
| **3** | 0.3300 | 1.2448 | **0.0000** | 1.4797 | 0.2000 | 0.9790 | **0.0400** | 0.9947 |

**Table 141. *t*-Test ACSE/DACSE - File 9 (10 Nodes)**

| *t*-Test & DACSE/ACSE Mean Ratio Results - 10 Node (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2100 | 0.9942 | 0.9100 | 0.9993 | 0.4400 | 0.9944 | 0.3100 | 1.1293 |
| **2** | 0.6500 | 0.9973 | 0.0800 | 1.0080 | 0.4500 | 0.9954 | 0.7300 | 0.9980 |
| **3** | 0.7200 | 0.9974 | 0.1500 | 1.0140 | 0.1800 | 1.0139 | 0.9500 | 0.9872 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7600 | 1.0021 | 0.6300 | 1.0019 | 0.8600 | 1.0012 | 0.6900 | 0.9980 |
| **2** | 0.3600 | 1.1233 | 0.8900 | 1.0003 | 0.9100 | 1.0007 | 0.3000 | 1.0058 |
| **3** | 1.0000 | 1.0001 | 0.2000 | 1.0060 | 0.3300 | 0.6517 | 0.2000 | 0.7284 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7000 | 0.9978 | 0.5200 | 1.1233 | 0.3300 | 0.9762 | **0.0000** | 1.0003 |
| **2** | 0.6100 | 0.9983 | **0.0300** | 0.5923 | 1.0000 | 1.0000 | **0.0100** | 1.0001 |
| **3** | 0.9800 | 0.9943 | 0.5700 | 1.1103 | 0.6700 | 0.9953 | 0.9400 | 1.0004 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6500 | 0.9985 | 0.5000 | 0.9970 | 0.9000 | 0.9639 | 0.6100 | 0.9354 |
| **2** | 0.8200 | 0.9992 | 0.7800 | 0.9985 | 0.6600 | 0.8835 | 0.8600 | 0.9825 |
| **3** | 0.5300 | 0.9962 | 0.1600 | 0.7153 | 0.5500 | 0.8443 | 0.7500 | 0.9369 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2600 | 1.0087 | 0.8800 | 1.0384 | 0.3100 | 1.0854 | 0.6500 | 1.0000 |
| **2** | 0.1600 | 0.3575 | 0.2100 | 0.7520 | 0.2400 | 1.0175 | 0.1500 | 1.0176 |
| **3** | 0.8700 | 0.9988 | 0.7700 | 1.0800 | 0.1000 | 1.0212 | 0.1500 | 1.0048 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0400** | 1.0105 | 0.4900 | 0.9200 | 0.7700 | 1.0000 | 1.0000 | 1.0000 |
| **2** | 0.3400 | 1.0043 | 0.3100 | 0.9301 | 0.4200 | 1.0001 | 1.0000 | 1.0000 |
| **3** | 0.3100 | 0.8860 | 0.8700 | 1.0250 | 0.5800 | 0.9999 | 1.0000 | 1.0000 |

**Table 142. *t*-Test ACSE/DACSE - File 10 (10 Nodes)**

| *t*-Test & DACSE/ACSE Mean Ratio Results - 10 Node (File 10) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7400 | 1.0001 | 0.7600 | 0.9999 | 0.1700 | 1.0084 | **0.0400** | 0.9994 |
| **2** | 0.6500 | 1.0001 | 0.2500 | 1.0003 | 0.3100 | 1.0044 | 0.3100 | 1.0044 |
| **3** | **0.0400** | 1.0006 | 0.2600 | 1.0050 | 0.5800 | 1.0002 | 0.1400 | 1.0091 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3100 | 1.0044 | 0.2000 | 1.0260 | 0.7800 | 0.9943 | **0.0000** | 0.9649 |
| **2** | 0.2000 | 1.0004 | 0.0500 | 0.9637 | 0.8300 | 0.9955 | 0.6300 | 0.9946 |
| **3** | 0.2200 | 0.9996 | 0.6200 | 0.9901 | 0.4600 | 0.9854 | **0.0000** | 0.9581 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2100 | 0.9996 | 0.2600 | 1.0045 | 0.1200 | 1.5085 | 0.3200 | 1.0001 |
| **2** | 0.4000 | 1.0002 | 0.6500 | 0.9975 | **0.0100** | 2.1300 | **0.0300** | 1.0003 |
| **3** | 0.2400 | 0.9995 | 0.7500 | 0.9990 | 0.4100 | 1.3534 | 0.3200 | 0.9764 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9400 | 1.0000 | 0.5600 | 0.9743 | 0.5300 | 1.1175 | **0.0100** | 0.9942 |
| **2** | 0.0500 | 0.9993 | 0.5700 | 1.0000 | **0.0400** | 1.2701 | 0.7500 | 1.0017 |
| **3** | 0.0700 | 1.0007 | 0.0500 | 1.1787 | 0.9700 | 1.0075 | 0.3300 | 0.9965 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2600 | 1.0049 | 0.0900 | 1.0003 | **0.0100** | 1.0067 | 0.7900 | 1.0000 |
| **2** | 0.1600 | 1.0714 | 0.1600 | 1.0713 | 0.1600 | 1.0723 | 0.1600 | 1.0715 |
| **3** | 0.4600 | 1.0003 | 0.8800 | 1.0000 | **0.0100** | 1.0077 | 0.4400 | 0.9999 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7200 | 0.9999 | 0.6300 | 1.1088 | 0.1600 | 1.0111 | 1.0000 | 1.0000 |
| **2** | 0.6800 | 1.0001 | 0.6500 | 1.1376 | 0.3600 | 0.9949 | 1.0000 | 1.0000 |
| **3** | 0.7700 | 0.9999 | 0.2900 | 1.2222 | 0.3400 | 0.9559 | 1.0000 | 1.0000 |

## Appendix H: *t*-Tests ACSS/DACSS (10 Nodes)

### Table 143. *t*-Test ACSS/DACSS - File 1 (10 Nodes)

| *t*-Test & DACSS/ACSS Mean Ratio Results - 10 Node (File 1) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7100 | 1.0030 | 0.7100 | 1.0033 | 0.0600 | 1.0179 | 0.6000 | 1.0053 |
| **2** | **0.0100** | 1.0202 | 0.2600 | 1.0103 | 0.0500 | 1.0156 | **0.0000** | 1.0191 |
| **3** | 0.2900 | 0.9882 | 0.2000 | 0.9872 | 0.9300 | 0.9991 | 0.5600 | 1.0051 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0600 | 1.0141 | 0.1400 | 1.0086 | 0.7600 | 1.0023 | **0.0400** | 1.0117 |
| **2** | 0.4100 | 1.0067 | **0.0100** | 1.0150 | 0.8800 | 0.9989 | 0.1200 | 1.0089 |
| **3** | 0.4700 | 0.9925 | **0.0100** | 1.0278 | 0.1000 | 1.0128 | **0.0200** | 1.0164 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9600 | 0.9995 | 0.2500 | 0.9923 | 0.5400 | 0.9133 | **0.0300** | 1.0002 |
| **2** | **0.0100** | 1.0191 | 0.0500 | 0.9861 | 0.7000 | 0.9580 | 0.7100 | 1.0000 |
| **3** | 0.4800 | 1.0104 | 0.0800 | 0.9900 | 0.5900 | 1.0912 | 0.0500 | 1.0002 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9900 | 1.0001 | 0.3600 | 1.0031 | 0.1100 | 0.9485 | 0.8800 | 0.9498 |
| **2** | **0.0300** | 1.0215 | 0.3300 | 0.9947 | 0.8000 | 0.9905 | 0.8600 | 1.0615 |
| **3** | 0.2900 | 1.0137 | 0.7300 | 1.0001 | 0.6100 | 0.9856 | 0.6900 | 1.0039 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7700 | 1.0033 | 0.2600 | 1.0036 | 0.8700 | 1.0000 | 0.6900 | 0.9969 |
| **2** | 0.1400 | 1.0095 | 0.8300 | 1.0007 | 0.8000 | 1.0000 | 0.6100 | 0.9962 |
| **3** | 0.4700 | 0.9925 | 0.5900 | 0.9978 | 0.1300 | 0.9997 | 0.2800 | 0.9959 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8700 | 0.9987 | 0.3900 | 1.0054 | 0.9400 | 1.0000 | 0.4700 | 0.9999 |
| **2** | 0.5900 | 1.0047 | 0.9500 | 1.0005 | 0.8200 | 1.0001 | **0.0300** | 1.0002 |
| **3** | 0.8600 | 1.0021 | 0.6900 | 1.0030 | 0.3800 | 0.9998 | 0.3900 | 1.0001 |

**Table 144. *t*-Test ACSS/DACSS - File 2 (10 Nodes)**

| *t*-Test & DACSS/ACSS Mean Ratio Results - 10 Node (File 2) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9600 | 0.9996 | 0.0700 | 1.0066 | 0.7000 | 1.0018 | 0.8700 | 1.0007 |
| **2** | 0.6900 | 1.0020 | 0.1900 | 1.0057 | 0.3600 | 1.0036 | 0.8700 | 1.0007 |
| **3** | 0.5700 | 1.0030 | 0.3300 | 1.0045 | 0.2000 | 1.0053 | **0.0300** | 1.0095 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0400** | 1.0091 | **0.0100** | 1.0091 | **0.0000** | 1.0159 | 0.3000 | 1.0047 |
| **2** | 0.3100 | 0.9956 | 0.1000 | 1.0063 | 0.7800 | 1.0017 | **0.0100** | 1.0081 |
| **3** | 0.1200 | 1.0064 | **0.0100** | 1.0127 | **0.0200** | 1.0147 | 0.1700 | 1.0069 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6600 | 1.0024 | 0.1100 | 1.0080 | 0.1200 | 0.9939 | 0.1400 | 0.9995 |
| **2** | 0.3100 | 1.0058 | **0.0000** | 1.0154 | 0.4200 | 0.9955 | 0.2000 | 0.9924 |
| **3** | 0.6100 | 1.0043 | 0.9600 | 1.0003 | 0.9200 | 0.9994 | 0.6700 | 0.9999 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7200 | 1.0011 | 0.9900 | 1.0001 | 0.4500 | 1.0050 | 0.6000 | 1.0069 |
| **2** | 0.1700 | 1.0051 | 0.0700 | 0.9886 | 0.1400 | 0.9887 | 0.9800 | 0.9995 |
| **3** | 0.6000 | 0.9975 | 0.5500 | 1.0057 | 0.4300 | 1.0055 | 0.6000 | 1.0112 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2400 | 1.0045 | 0.7900 | 1.0054 | 0.7900 | 0.9890 | 0.3800 | 0.9859 |
| **2** | 0.1400 | 1.0067 | **0.0200** | 1.0491 | 0.2800 | 0.9531 | 0.2300 | 0.9746 |
| **3** | 0.2100 | 0.9916 | 0.2500 | 1.0451 | 0.3900 | 1.0566 | 0.2900 | 1.0218 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8200 | 1.0010 | 0.7200 | 1.0018 | 0.9100 | 0.9993 | 0.8100 | 0.9940 |
| **2** | 0.1300 | 1.0072 | 0.6100 | 1.0023 | 0.2600 | 0.9357 | 0.0800 | 0.9441 |
| **3** | 0.5800 | 1.0029 | 0.5900 | 1.0025 | 0.8500 | 1.0015 | 0.3600 | 0.9805 |

Table 145. *t*-Test ACSS/DACSS - File 3 (10 Nodes)

| *t*-Test & DACSS/ACSS Mean Ratio Results - 10 Node (File 3) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4500 | 0.9970 | 0.7300 | 0.9986 | **0.0200** | 0.9926 | 0.1300 | 0.9956 |
| **2** | 0.2900 | 0.9958 | 0.5000 | 0.9967 | 0.6200 | 0.9981 | 0.2200 | 0.9947 |
| **3** | 0.8500 | 0.9991 | 0.3100 | 0.9944 | 0.0700 | 0.9909 | 0.0900 | 0.9936 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5300 | 0.9968 | 0.9000 | 1.0007 | 0.7200 | 1.0016 | 0.4000 | 1.0031 |
| **2** | 0.3700 | 0.9963 | 0.2500 | 1.0056 | 0.3300 | 1.0043 | 0.3000 | 0.9951 |
| **3** | 0.2200 | 0.9917 | 0.8500 | 1.0013 | 0.3700 | 1.0043 | 0.0500 | 0.9922 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9900 | 1.0000 | 0.1200 | 1.0048 | 0.3900 | 1.0001 | 0.5300 | 1.0001 |
| **2** | 0.0800 | 1.0064 | 0.3900 | 0.9972 | 0.1200 | 1.0001 | 0.3500 | 0.9999 |
| **3** | 0.6600 | 0.9971 | 0.5400 | 1.0029 | 0.8700 | 1.0000 | 0.1100 | 1.0002 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1500 | 0.9954 | 1.0000 | 1.0000 | 0.7100 | 1.0014 | 0.4200 | 1.0001 |
| **2** | 0.2400 | 0.9944 | 0.2000 | 1.0039 | 0.4100 | 0.9961 | 0.8500 | 1.0000 |
| **3** | 0.4800 | 1.0031 | 0.6500 | 0.9982 | 0.8300 | 0.9989 | 0.2800 | 1.0002 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0400** | 0.9910 | 0.2000 | 1.0048 | 0.6200 | 0.9967 | 0.9900 | 0.9999 |
| **2** | 0.8000 | 1.0008 | 0.9600 | 0.9998 | 0.5300 | 0.9954 | **0.0100** | 1.0133 |
| **3** | 0.9800 | 0.9999 | 0.5900 | 0.9971 | 0.0600 | 0.9878 | 0.1000 | 0.9941 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3100 | 0.9958 | 0.3100 | 0.9974 | 0.1200 | 1.0004 | 0.4700 | 1.0036 |
| **2** | 0.5600 | 0.9975 | **0.0200** | 0.9931 | **0.0100** | 1.0007 | 0.3600 | 0.9927 |
| **3** | 0.3700 | 0.9945 | **0.0400** | 0.9931 | 0.0500 | 1.0007 | 0.4200 | 0.9955 |

## Table 146. *t*-Test ACSS/DACSS - File 4 (10 Nodes)

| ***t*-Test & DACSS/ACSS Mean Ratio Results - 10 Node (File 4)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7000 | 1.0017 | 0.4800 | 1.0035 | 0.6800 | 1.0019 | **0.0300** | 1.0083 |
| **2** | 0.5300 | 1.0024 | 0.7000 | 1.0020 | 0.4400 | 1.0034 | 0.8300 | 0.9991 |
| **3** | 0.0900 | 1.0108 | 0.6900 | 1.0019 | 0.1800 | 1.0058 | 0.3900 | 1.0040 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3600 | 0.9959 | 0.0600 | 1.0100 | **0.0000** | 1.0159 | 0.3400 | 1.0032 |
| **2** | 0.5500 | 0.9978 | 0.0900 | 1.0084 | **0.0000** | 1.0117 | **0.0100** | 1.0081 |
| **3** | 0.9000 | 1.0008 | **0.0400** | 1.0099 | **0.0200** | 1.0105 | 0.7400 | 0.9989 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9400 | 0.9995 | 0.3000 | 0.9936 | 0.0700 | 1.0087 | 0.1800 | 1.0006 |
| **2** | **0.0300** | 1.0107 | 0.0500 | 0.9885 | 0.8200 | 1.0009 | 0.1200 | 1.0007 |
| **3** | 0.8600 | 1.0012 | 0.4100 | 1.0052 | 0.1600 | 0.9924 | 0.0600 | 1.0008 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0200** | 1.0119 | 0.1400 | 0.9870 | 0.5900 | 0.9886 | 0.2200 | 1.0189 |
| **2** | 0.0500 | 1.0101 | 0.1700 | 0.9877 | 0.4300 | 0.9863 | **0.0100** | 1.0429 |
| **3** | 0.3500 | 1.0048 | 0.2900 | 0.9912 | 0.5700 | 0.9462 | **0.0300** | 1.0337 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5800 | 0.9936 | 0.2300 | 1.0865 | 0.6400 | 1.0030 | **0.0400** | 1.0419 |
| **2** | 0.1900 | 1.0053 | 0.7300 | 0.9825 | 0.2100 | 0.9907 | 0.4700 | 0.9853 |
| **3** | 0.3300 | 1.0045 | 0.7500 | 1.0289 | 0.5200 | 1.0055 | 0.6800 | 0.9950 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1600 | 1.0059 | 0.1200 | 0.9956 | **0.0300** | 0.9999 | **0.0100** | 0.9752 |
| **2** | 0.4700 | 1.0038 | 0.6600 | 1.0011 | 0.5400 | 1.0000 | **0.0100** | 0.9705 |
| **3** | 0.2000 | 1.0055 | 0.8700 | 1.0006 | 0.8100 | 0.9989 | 0.2800 | 0.9873 |

**Table 147. *t*-Test ACSS/DACSS - File 5 (10 Nodes)**

| *t*-Test & DACSS/ACSS Mean Ratio Results - 10 Node (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1200 | 1.0069 | **0.0200** | 1.0093 | **0.0100** | 1.0129 | 0.1200 | 1.0079 |
| **2** | 0.1800 | 1.0060 | **0.0000** | 1.0147 | **0.0000** | 1.0147 | **0.0400** | 1.0091 |
| **3** | **0.0000** | 1.0193 | 0.4100 | 1.0047 | **0.0000** | 1.0157 | 0.1500 | 1.0081 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0100** | 1.0122 | **0.0200** | 1.0136 | 0.1100 | 1.0060 | 0.2300 | 1.0037 |
| **2** | **0.0000** | 1.0158 | 0.2200 | 1.0076 | 0.1800 | 1.0066 | 0.2800 | 1.0045 |
| **3** | 0.6300 | 0.9940 | **0.0200** | 1.0132 | 0.4000 | 1.0037 | **0.0000** | 1.0098 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0100** | 1.0125 | 0.6900 | 1.0015 | 0.9800 | 1.0004 | 0.4600 | 1.0022 |
| **2** | **0.0100** | 1.0134 | **0.0100** | 1.0110 | 0.3200 | 0.9920 | 0.3500 | 1.0026 |
| **3** | **0.0000** | 1.0233 | 0.0500 | 1.0106 | 0.1100 | 0.9479 | 0.3300 | 0.9972 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0200** | 1.0117 | **0.0000** | 1.0223 | 0.1300 | 0.9901 | 0.0500 | 1.0011 |
| **2** | **0.0300** | 1.0129 | **0.0300** | 1.0151 | 0.8100 | 0.9985 | 0.1600 | 1.0009 |
| **3** | 0.1700 | 1.0099 | 0.5200 | 1.0039 | 0.9100 | 0.9994 | 0.7300 | 1.0002 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0000** | 1.0174 | 0.8100 | 0.9990 | **0.0000** | 0.9995 | 0.4100 | 1.0000 |
| **2** | **0.0100** | 1.0131 | 0.6600 | 0.9979 | **0.0100** | 0.9997 | 0.2900 | 1.0000 |
| **3** | **0.0400** | 1.0196 | 0.2600 | 1.0090 | 0.3000 | 1.0002 | 0.2200 | 1.0000 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4200 | 1.0035 | 0.1300 | 0.9996 | 0.1100 | 1.0000 | 0.6500 | 1.0000 |
| **2** | 0.1100 | 1.0067 | 0.2800 | 0.9998 | 0.0500 | 1.0001 | 0.0600 | 1.0000 |
| **3** | **0.0400** | 1.0114 | 0.2800 | 0.9998 | **0.0000** | 1.0001 | 0.6000 | 1.0000 |

**Table 148. *t*-Test ACSS/DACSS - File 6 (10 Nodes)**

| ***t*-Test & DACSS/ACSS Mean Ratio Results - 10 Node (File 6)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9000 | 0.9995 | 0.1000 | 0.9923 | 0.5100 | 0.9966 | 0.1800 | 0.9938 |
| **2** | 0.5400 | 1.0028 | 0.6800 | 1.0023 | 0.6300 | 0.9975 | 0.2600 | 1.0053 |
| **3** | **0.0400** | 0.9909 | 0.9900 | 0.9999 | 0.1700 | 0.9940 | 0.2700 | 0.9950 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6500 | 1.0023 | 0.7500 | 1.0010 | **0.0200** | 1.0089 | 0.3200 | 0.9955 |
| **2** | 0.5000 | 0.9970 | 0.1500 | 1.0054 | 0.1000 | 1.0059 | 0.6500 | 1.0019 |
| **3** | 0.9500 | 1.0002 | 0.7300 | 0.9985 | **0.0000** | 1.0165 | 0.1800 | 1.0067 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1500 | 0.9939 | 0.6400 | 0.9976 | 0.7800 | 0.9992 | 0.7100 | 1.0000 |
| **2** | 0.1300 | 0.9926 | 0.4400 | 1.0037 | 0.2300 | 0.9964 | **0.0400** | 1.0002 |
| **3** | 0.7200 | 0.9982 | 0.4300 | 0.9946 | 0.7200 | 1.0028 | **0.0100** | 1.0002 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0100** | 0.9872 | 0.2700 | 0.9953 | 0.6300 | 0.9973 | 0.5000 | 0.9974 |
| **2** | 0.0500 | 0.9907 | 0.5500 | 0.9975 | 0.2400 | 0.9934 | 0.3600 | 0.9967 |
| **3** | **0.0100** | 0.9853 | 0.2200 | 0.9949 | 0.2000 | 0.9911 | 0.3600 | 0.9964 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5900 | 0.9976 | 0.3700 | 1.0038 | 0.5400 | 1.0041 | 0.1700 | 1.0000 |
| **2** | 0.7700 | 0.9986 | 0.4900 | 0.9974 | 0.5800 | 0.9956 | 0.5200 | 1.0000 |
| **3** | 0.2200 | 0.9912 | 0.7300 | 1.0019 | 0.3800 | 0.9915 | 0.6900 | 1.0000 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1900 | 0.9940 | 0.9100 | 1.0007 | 0.6500 | 0.9996 | 1.0000 | 1.0000 |
| **2** | 0.1600 | 1.0069 | 0.8800 | 0.9990 | 0.8900 | 0.9999 | 1.0000 | 1.0000 |
| **3** | 0.5800 | 1.0038 | 0.1100 | 1.0168 | 0.1900 | 0.9947 | 1.0000 | 1.0000 |

**Table 149. *t*-Test ACSS/DACSS - File 7 (10 Nodes)**

| ***t*-Test & DACSS/ACSS Mean Ratio Results - 10 Node (File 7)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0100** | 1.0138 | 0.7000 | 1.0025 | 0.9800 | 1.0001 | 0.8200 | 1.0011 |
| **2** | 0.7500 | 1.0015 | 0.9400 | 1.0004 | 0.5400 | 0.9975 | 0.4100 | 1.0035 |
| **3** | 0.8000 | 1.0013 | 0.2400 | 1.0059 | **0.0200** | 1.0126 | 0.8200 | 0.9990 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5300 | 1.0043 | **0.0300** | 0.9915 | 0.9400 | 1.0004 | 0.2400 | 1.0114 |
| **2** | 0.2500 | 0.9921 | 0.1700 | 0.9957 | 0.1900 | 1.0072 | 0.5800 | 1.0019 |
| **3** | 0.4200 | 1.0063 | 0.0900 | 0.9927 | 0.4500 | 0.9961 | 0.8700 | 0.9993 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9600 | 0.9998 | 0.6700 | 1.0000 | **0.0000** | 1.0008 | **0.0000** | 1.0006 |
| **2** | 0.2900 | 0.9961 | **0.0000** | 1.0013 | 0.8400 | 1.0000 | 0.0700 | 1.0003 |
| **3** | 0.9500 | 1.0003 | 0.6800 | 1.0000 | 0.2700 | 1.0002 | **0.0200** | 1.0005 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0100** | 0.9800 | 0.1300 | 1.0054 | 0.4100 | 1.0037 | 0.9900 | 0.9999 |
| **2** | 0.9200 | 1.0007 | 0.5500 | 1.0025 | 0.6800 | 1.0016 | 0.3700 | 0.9935 |
| **3** | 0.3400 | 0.9924 | 0.5900 | 0.9975 | 0.2700 | 1.0054 | 0.0700 | 0.9893 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2000 | 1.0096 | 0.6200 | 0.9980 | 0.3400 | 0.8982 | 0.9400 | 1.0014 |
| **2** | 0.1700 | 1.0064 | 0.3700 | 0.9973 | 0.7000 | 0.9525 | 0.1400 | 1.0349 |
| **3** | **0.0000** | 1.0125 | 0.1000 | 1.0448 | 0.8700 | 1.0259 | 0.2200 | 0.9931 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8200 | 0.9990 | 0.6400 | 0.9979 | 0.1500 | 1.0002 | 0.1500 | 1.0000 |
| **2** | **0.0400** | 1.0100 | 0.7300 | 0.9986 | 0.8400 | 1.0000 | 1.0000 | 1.0000 |
| **3** | 0.7400 | 1.0018 | 0.6500 | 0.9975 | 0.3500 | 1.0001 | 0.2800 | 1.0000 |

**Table 150. *t*-Test ACSS/DACSS - File 8 (10 Nodes)**

| *t*-Test & DACSS/ACSS Mean Ratio Results - 10 Node (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0000** | 1.0113 | **0.0400** | 1.0067 | 0.0800 | 1.0070 | **0.0000** | 1.0126 |
| **2** | 0.0800 | 1.0071 | **0.0000** | 1.0131 | **0.0000** | 1.0135 | **0.0200** | 1.0087 |
| **3** | **0.0000** | 1.0118 | 0.0600 | 1.0083 | **0.0100** | 1.0099 | **0.0100** | 1.0110 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0400** | 1.0070 | **0.0100** | 1.0122 | 0.5000 | 1.0001 | 0.0500 | 0.9999 |
| **2** | 0.2100 | 1.0043 | **0.0100** | 1.0129 | 0.1100 | 1.0001 | **0.0000** | 0.9998 |
| **3** | 0.3200 | 1.0039 | **0.0000** | 1.0169 | 0.4600 | 0.9999 | **0.0000** | 0.9998 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0000** | 1.0082 | 0.1600 | 0.9880 | 0.9900 | 1.0016 | 0.1000 | 0.9997 |
| **2** | 0.1800 | 1.0056 | 0.5500 | 0.9955 | 0.4400 | 1.0043 | 0.1400 | 1.0004 |
| **3** | **0.0100** | 1.0094 | 0.2500 | 1.0141 | 0.8400 | 0.9976 | 0.8600 | 1.0000 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0100** | 1.0120 | 0.5800 | 1.0020 | **0.0100** | 1.0144 | 0.0900 | 0.9821 |
| **2** | 0.5700 | 1.0019 | 0.5300 | 1.0023 | 0.0800 | 1.0116 | 0.4900 | 0.9789 |
| **3** | 0.8700 | 1.0007 | 0.8000 | 1.0013 | 0.3400 | 0.9912 | 0.9700 | 0.9995 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0000** | 1.0149 | **0.0200** | 1.0056 | 0.2100 | 1.0039 | 0.6200 | 0.9986 |
| **2** | 0.0700 | 1.0054 | **0.0100** | 1.0064 | 0.8400 | 1.0007 | 0.6100 | 1.0008 |
| **3** | **0.0100** | 1.0087 | **0.0000** | 1.0104 | 0.7700 | 0.9992 | 0.4900 | 1.0028 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8100 | 1.0009 | 0.9300 | 0.9993 | 0.3000 | 0.9996 | 0.2700 | 1.0058 |
| **2** | 0.7500 | 1.0011 | 0.7000 | 0.9957 | 0.1600 | 0.9996 | 0.3000 | 1.0050 |
| **3** | 0.8300 | 0.9991 | 0.6100 | 1.0045 | **0.0100** | 0.9990 | **0.0000** | 1.0298 |

## Table 151. *t*-Test ACSS/DACSS - File 9 (10 Nodes)

| *t*-Test & DACSS/ACSS Mean Ratio Results - 10 Node (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3900 | 0.9995 | 0.4200 | 0.9923 | 0.5700 | 0.9966 | 0.3700 | 0.9938 |
| **2** | 0.6800 | 1.0028 | 0.3300 | 1.0023 | 0.5800 | 0.9975 | 0.4700 | 1.0053 |
| **3** | 0.0500 | 0.9909 | 0.4700 | 0.9999 | 0.9100 | 0.9940 | 0.3600 | 0.9950 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0900 | 1.0023 | 0.1400 | 1.0010 | 0.1400 | 1.0089 | 0.0700 | 0.9955 |
| **2** | 0.5200 | 0.9970 | 0.2500 | 1.0054 | 0.5700 | 1.0059 | 0.0600 | 1.0019 |
| **3** | 0.2600 | 1.0002 | **0.0200** | 0.9985 | 0.2400 | 1.0165 | 0.6400 | 1.0067 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1900 | 0.9939 | 0.3800 | 0.9976 | 0.8700 | 0.9992 | **0.0300** | 1.0000 |
| **2** | 0.2900 | 0.9926 | 0.5800 | 1.0037 | 0.0700 | 0.9964 | 0.3500 | 1.0002 |
| **3** | 0.7200 | 0.9982 | 0.3500 | 0.9946 | 0.1500 | 1.0028 | 0.1000 | 1.0002 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2200 | 0.9872 | 0.4700 | 0.9953 | 0.2100 | 0.9973 | **0.0200** | 0.9974 |
| **2** | 0.3500 | 0.9907 | 0.4900 | 0.9975 | 0.1600 | 0.9934 | **0.0100** | 0.9967 |
| **3** | 0.3700 | 0.9853 | 0.6700 | 0.9949 | 0.9000 | 0.9911 | **0.0000** | 0.9964 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3000 | 0.9976 | 0.0500 | 1.0038 | 0.4600 | 1.0041 | 0.6400 | 1.0000 |
| **2** | 0.1200 | 0.9986 | 0.7200 | 0.9974 | **0.0100** | 0.9956 | 0.8500 | 1.0000 |
| **3** | 0.0600 | 0.9912 | 0.4000 | 1.0019 | **0.0000** | 0.9915 | **0.0000** | 1.0000 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0300** | 0.9940 | 0.2300 | 1.0007 | 0.1700 | 0.9996 | 0.2000 | 1.0000 |
| **2** | 0.2700 | 1.0069 | 0.5400 | 0.9990 | 0.3800 | 0.9999 | 0.6300 | 1.0000 |
| **3** | 0.2100 | 1.0038 | 0.6800 | 1.0168 | 0.1700 | 0.9947 | 0.3300 | 1.0000 |

**Table 152. *t*-Test ACSS/DACSS - File 10 (10 Nodes)**

| ***t*-Test & DACSS/ACSS Mean Ratio Results - 10 Node (File 10)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2400 | 0.9995 | 0.2900 | 0.9923 | 0.7900 | 0.9966 | 0.3100 | 0.9938 |
| **2** | 0.4100 | 1.0028 | 0.6100 | 1.0023 | 0.0800 | 0.9975 | 0.3000 | 1.0053 |
| **3** | 0.1800 | 0.9909 | 0.7300 | 0.9999 | 0.3300 | 0.9940 | 0.5400 | 0.9950 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4400 | 1.0023 | 0.0800 | 1.0010 | **0.0100** | 1.0089 | **0.0400** | 0.9955 |
| **2** | 0.9800 | 0.9970 | **0.0100** | 1.0054 | **0.0100** | 1.0059 | 0.2500 | 1.0019 |
| **3** | **0.0300** | 1.0002 | 0.1100 | 0.9985 | **0.0000** | 1.0165 | 0.4800 | 1.0067 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8700 | 0.9939 | 0.3500 | 0.9976 | 0.6000 | 0.9992 | **0.0400** | 1.0000 |
| **2** | 0.9800 | 0.9926 | 0.2100 | 1.0037 | 0.0800 | 0.9964 | 0.0600 | 1.0002 |
| **3** | 0.0600 | 0.9982 | 0.7400 | 0.9946 | 0.4700 | 1.0028 | **0.0200** | 1.0002 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0000** | 0.9872 | 0.6200 | 0.9953 | 0.5200 | 0.9973 | **0.0000** | 0.9974 |
| **2** | 0.6200 | 0.9907 | **0.0300** | 0.9975 | 0.9900 | 0.9934 | **0.0200** | 0.9967 |
| **3** | **0.0300** | 0.9853 | 0.0700 | 0.9949 | 0.7400 | 0.9911 | 0.0900 | 0.9964 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9900 | 0.9976 | **0.0000** | 1.0038 | **0.0000** | 1.0041 | 0.1800 | 1.0000 |
| **2** | 0.0700 | 0.9986 | **0.0200** | 0.9974 | **0.0400** | 0.9956 | 0.2800 | 1.0000 |
| **3** | 0.3300 | 0.9912 | 0.0600 | 1.0019 | 0.5800 | 0.9915 | 0.3200 | 1.0000 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7400 | 0.9940 | 0.2900 | 1.0007 | 0.5100 | 0.9996 | 0.1500 | 1.0000 |
| **2** | 0.7800 | 1.0069 | 0.9100 | 0.9990 | 0.5000 | 0.9999 | 0.0500 | 1.0000 |
| **3** | 0.5300 | 1.0038 | **0.0400** | 1.0168 | 0.8900 | 0.9947 | 0.9300 | 1.0000 |

## Appendix I: *t*-Tests ACSE/DACSE (15 Nodes)

### Table 153. *t*-Test ACSE/DACSE - File 1 (15 Nodes)

| ***t*-Test & DACSE/ACSE Mean Ratio Results - 15 Node (File 1)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4300 | 0.8450 | 0.1400 | 1.3211 | 0.9100 | 1.0277 | 0.6800 | 0.9066 |
| **2** | 0.4200 | 1.1711 | 0.2800 | 1.2468 | 0.2700 | 0.8009 | 0.4800 | 1.1527 |
| **3** | 0.3500 | 0.8184 | 0.8800 | 1.0310 | 0.2900 | 1.2897 | 0.9600 | 1.0108 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2700 | 0.7882 | 0.3000 | 0.8137 | 0.1100 | 1.2187 | 0.5600 | 1.0954 |
| **2** | 0.7200 | 0.9196 | 0.9800 | 1.0051 | 1.0000 | 0.9995 | 0.1100 | 0.7489 |
| **3** | 0.4000 | 1.1408 | 0.3900 | 0.7924 | **0.0000** | 1.3807 | 0.9400 | 0.9869 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6500 | 1.0894 | 0.2300 | 0.8888 | 0.8200 | 0.9712 | 0.6500 | 0.8954 |
| **2** | 0.9200 | 0.9807 | 0.5900 | 0.9556 | 0.2300 | 1.1075 | 0.8000 | 0.9470 |
| **3** | 0.4500 | 0.8538 | 0.1000 | 1.2206 | 0.0900 | 1.2621 | 0.8800 | 0.9621 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9500 | 0.9881 | 0.5100 | 0.9091 | 0.8800 | 0.9961 | 0.4100 | 1.1035 |
| **2** | 0.6800 | 1.0723 | 0.0500 | 0.7678 | 0.2500 | 1.0122 | 0.2400 | 0.8858 |
| **3** | 0.4500 | 1.1399 | 0.1000 | 0.7600 | 0.9100 | 1.0143 | 0.8700 | 1.0199 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2500 | 1.2369 | 0.7100 | 1.0152 | 0.3500 | 1.0601 | 0.6200 | 1.0050 |
| **2** | 0.4800 | 1.1319 | 0.8500 | 1.0097 | 0.1500 | 1.0912 | 0.3200 | 1.0096 |
| **3** | 0.5600 | 0.8990 | 0.5400 | 1.0275 | 0.7200 | 1.0201 | 0.7500 | 1.0041 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5900 | 1.1302 | 0.2300 | 1.2575 | 0.9600 | 1.0081 | 0.1500 | 0.8126 |
| **2** | 0.2000 | 0.7475 | 0.7800 | 1.0498 | 0.0800 | 1.1854 | 0.0500 | 0.6494 |
| **3** | 0.2000 | 1.2647 | 0.6400 | 0.9064 | 0.2900 | 0.8067 | 0.5600 | 0.9174 |

Table 154. *t*-Test ACSE/DACSE - File 2 (15 Nodes)

| *t*-Test & DACSE/ACSE Mean Ratio Results - 15 Node (File 2) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4000 | 0.8735 | 0.3000 | 0.8103 | **0.0200** | 0.7070 | 0.0600 | 1.3546 |
| **2** | **0.0200** | 0.6623 | 0.9300 | 1.0129 | 0.4200 | 0.8959 | 0.9700 | 1.0056 |
| **3** | 0.2500 | 0.8259 | 0.7900 | 0.9562 | 0.5700 | 1.1172 | 0.4400 | 1.1558 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9900 | 1.0021 | 0.2800 | 0.9220 | 0.5800 | 0.9462 | 0.4300 | 0.9266 |
| **2** | 0.5300 | 0.9140 | 0.1600 | 0.8815 | 0.8400 | 0.9800 | 0.4700 | 0.9260 |
| **3** | **0.0400** | 0.6972 | 0.5700 | 1.0972 | 0.3500 | 1.1390 | 0.7700 | 1.0510 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9500 | 1.0100 | 0.1800 | 0.8696 | 0.7300 | 1.0558 | 0.4200 | 1.1387 |
| **2** | 0.2000 | 0.8281 | 0.4100 | 1.0877 | 0.5600 | 0.9350 | 0.2800 | 0.8439 |
| **3** | 0.4400 | 1.1454 | 0.1000 | 1.1841 | 0.1600 | 0.7772 | 0.2700 | 0.8625 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1900 | 1.1983 | 0.1300 | 0.9602 | 0.8100 | 0.9873 | 0.6400 | 0.9753 |
| **2** | 0.8000 | 1.0400 | 0.6500 | 0.9910 | 0.3600 | 1.0468 | 0.9300 | 1.0049 |
| **3** | 0.7400 | 0.9429 | 0.9500 | 1.0026 | 0.8400 | 0.9907 | 0.6200 | 1.0262 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0600 | 1.3114 | 0.9600 | 0.9967 | 0.7200 | 1.0258 | **0.0200** | 1.1286 |
| **2** | 0.4200 | 0.8878 | 0.5100 | 0.9390 | 0.4300 | 1.0597 | 0.7700 | 1.0088 |
| **3** | 0.1000 | 0.7408 | 0.5300 | 1.0835 | **0.0100** | 1.1891 | 0.5600 | 1.0283 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1300 | 0.7881 | 0.5100 | 0.8816 | 0.4000 | 0.9795 | 0.3400 | 0.9791 |
| **2** | 0.2600 | 0.8538 | 0.1700 | 0.8070 | 0.0500 | 1.1631 | 0.4700 | 1.0166 |
| **3** | 0.2200 | 1.2679 | 0.0800 | 1.3933 | 0.8100 | 0.9926 | 0.5500 | 1.0192 |

**Table 155. *t*-Test ACSE/DACSE - File 3 (15 Nodes)**

| ***t*-Test & DACSE/ACSE Mean Ratio Results - 15 Node (File 3)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3600 | 1.1725 | 0.2600 | 1.2078 | 0.6100 | 0.9117 | 0.3300 | 1.1587 |
| **2** | 0.5600 | 1.1176 | 0.4800 | 1.1220 | 0.1400 | 0.7561 | 0.7100 | 0.9400 |
| **3** | 0.1000 | 0.7370 | 0.8700 | 1.0303 | 0.4100 | 1.1802 | 0.2600 | 0.8079 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1500 | 1.2774 | 0.8400 | 0.9750 | 0.0900 | 1.2145 | 0.3800 | 1.1116 |
| **2** | 0.9800 | 0.9963 | 0.6100 | 0.9424 | 0.1800 | 1.1857 | 0.2100 | 1.1716 |
| **3** | 0.9100 | 1.0212 | 0.6800 | 0.9534 | 0.1800 | 1.1820 | 0.1200 | 0.8301 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6100 | 1.0907 | 0.3800 | 0.9581 | 0.4700 | 0.9740 | 0.3800 | 1.0363 |
| **2** | 0.2700 | 0.8399 | 0.5500 | 1.0621 | 0.9100 | 0.9960 | 0.2700 | 0.9490 |
| **3** | 0.2600 | 0.8456 | 0.0700 | 1.1947 | 0.5400 | 0.9646 | 0.9300 | 1.0037 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9000 | 0.9800 | 0.7600 | 1.0673 | 0.1900 | 1.2210 | **0.0400** | 0.8596 |
| **2** | 0.0800 | 1.3892 | 0.6500 | 0.9081 | 0.6400 | 0.9275 | **0.0000** | 0.8214 |
| **3** | 0.9100 | 1.0231 | 0.5600 | 1.0871 | 0.7200 | 0.9410 | 0.6700 | 0.9710 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3300 | 1.1894 | 0.6200 | 0.9478 | 0.5800 | 0.9686 | 0.8700 | 1.0060 |
| **2** | 0.2000 | 1.2674 | 0.2700 | 1.1210 | 0.2300 | 0.9316 | 0.1600 | 1.0452 |
| **3** | 0.1200 | 0.7697 | 0.4300 | 1.0994 | 0.8300 | 0.9896 | 0.7100 | 0.9863 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6900 | 1.0694 | 0.6000 | 1.1104 | 0.0500 | 1.0978 | 0.7300 | 1.0125 |
| **2** | 0.2800 | 0.8189 | 0.6100 | 1.1024 | 0.4600 | 1.0304 | 0.1300 | 0.9684 |
| **3** | 0.4400 | 1.1487 | 0.3600 | 1.1814 | 0.8600 | 1.0092 | 0.6200 | 1.0169 |

**Table 156. *t*-Test ACSE/DACSE - File 4 (15 Nodes)**

| *t*-Test & DACSE/ACSE Mean Ratio Results - 15 Node (File 4) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3300 | 1.0368 | 0.7400 | 1.0074 | 0.2800 | 1.0193 | 0.1600 | 1.0248 |
| **2** | 0.1400 | 1.0757 | 0.4700 | 1.0375 | 0.1500 | 0.9450 | 0.1500 | 1.0614 |
| **3** | **0.0100** | 1.2703 | 0.1900 | 0.9625 | 0.3200 | 1.0312 | 0.4700 | 1.0215 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7200 | 0.9819 | 0.0500 | 1.1151 | 0.5900 | 0.9825 | 0.4300 | 0.9758 |
| **2** | **0.0100** | 1.1312 | 0.1000 | 1.1145 | 0.8800 | 0.9941 | 0.6300 | 1.0190 |
| **3** | 0.1700 | 0.9283 | 0.1800 | 1.0668 | 0.9200 | 1.0036 | 0.6100 | 0.9771 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2400 | 1.0526 | 0.1200 | 0.9389 | 0.5700 | 1.0405 | 0.2300 | 1.1594 |
| **2** | 0.0600 | 1.0916 | 0.4900 | 0.9687 | 0.3200 | 1.0805 | 0.7200 | 1.0355 |
| **3** | 0.5000 | 0.9677 | 0.0600 | 0.9143 | 0.6700 | 0.9691 | 0.7800 | 0.9623 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7100 | 1.0148 | 0.4300 | 0.9778 | 0.2800 | 0.9696 | 0.0900 | 0.9522 |
| **2** | **0.0400** | 1.0757 | **0.0000** | 1.0896 | 0.8400 | 0.9960 | 0.1900 | 1.0197 |
| **3** | 0.4200 | 1.0643 | 0.3800 | 1.0235 | 1.0000 | 1.0000 | 0.1200 | 0.9505 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4300 | 1.0310 | 0.9500 | 1.0065 | 0.1700 | 1.2993 | 0.3700 | 1.0358 |
| **2** | 0.6500 | 0.9759 | 0.2200 | 1.1785 | 0.3200 | 1.1562 | **0.0400** | 1.0614 |
| **3** | 0.7300 | 0.9826 | 0.7100 | 0.9536 | **0.0300** | 0.7383 | 0.9100 | 0.9929 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6800 | 1.0217 | **0.0000** | 0.5452 | 0.9600 | 0.9980 | 0.3600 | 0.9882 |
| **2** | 0.1900 | 1.0747 | 0.6400 | 0.9258 | 0.3400 | 1.0350 | 0.3900 | 0.9872 |
| **3** | **0.0400** | 1.1612 | 0.0600 | 1.3311 | 0.3400 | 0.9668 | 0.9300 | 0.9987 |

## Table 157. *t*-Test ACSE/DACSE - File 5 (15 Nodes)

| *t*-Test & DACSE/ACSE Mean Ratio Results - 15 Node (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8000 | 0.9561 | 0.5700 | 1.1388 | 0.2200 | 1.2815 | 0.8500 | 1.0403 |
| **2** | 0.0900 | 1.3889 | 0.0700 | 0.6964 | 0.1000 | 1.3717 | 0.2000 | 0.7959 |
| **3** | 0.9600 | 0.9897 | 0.7900 | 0.9144 | 0.2400 | 1.5426 | 0.1400 | 1.7209 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5400 | 1.1025 | 0.2900 | 0.8847 | 0.6200 | 1.0874 | 0.1400 | 0.8429 |
| **2** | 0.9100 | 1.0226 | 0.7800 | 0.9699 | 0.6800 | 1.0851 | 0.7000 | 0.9600 |
| **3** | 0.9600 | 0.9876 | 0.2000 | 0.8105 | 0.4700 | 1.1512 | 0.1800 | 0.7804 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7200 | 0.9361 | 0.3500 | 0.9059 | **0.0000** | 0.5209 | 0.9400 | 0.9955 |
| **2** | 0.1600 | 1.3149 | 0.3100 | 0.9006 | 0.3300 | 0.8187 | 0.8800 | 1.0080 |
| **3** | 0.9400 | 1.0170 | 0.7700 | 0.9617 | 0.3900 | 0.8786 | 0.8700 | 0.9909 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6100 | 1.0943 | 0.3900 | 0.8927 | 0.3700 | 1.0589 | 0.7900 | 0.9805 |
| **2** | 0.5500 | 1.1229 | 0.6400 | 1.0835 | 0.4900 | 1.0371 | 0.9500 | 0.9956 |
| **3** | 0.4100 | 1.1717 | 0.1000 | 1.3311 | 0.2200 | 1.0800 | 0.6000 | 1.0448 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0500 | 1.3912 | 0.4400 | 1.1767 | 0.3900 | 0.8379 | 0.4200 | 0.9476 |
| **2** | 0.7900 | 1.0583 | 0.2600 | 1.2022 | **0.0100** | 0.5671 | 0.9300 | 1.0044 |
| **3** | 0.5900 | 1.1029 | 0.1400 | 1.2943 | 0.3600 | 0.8340 | **0.0300** | 0.8620 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7600 | 0.9339 | 0.8900 | 0.9946 | 0.2400 | 0.9210 | 0.4600 | 1.0061 |
| **2** | 0.9000 | 0.9766 | **0.0300** | 0.9071 | 0.8000 | 1.0148 | 0.6400 | 0.9941 |
| **3** | 0.2000 | 0.8270 | **0.0400** | 1.1503 | 0.2300 | 0.9166 | 0.2000 | 0.9891 |

## Table 158. *t*-Test ACSE/DACSE - File 6 (15 Nodes)

| *t*-Test & DACSE/ACSE Mean Ratio Results - 15 Node (File 6) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2700 | 0.8401 | 0.8100 | 0.9640 | 0.1600 | 0.8306 | 0.2100 | 1.2361 |
| **2** | 0.2000 | 1.2374 | 0.9200 | 1.0165 | 0.4600 | 1.1272 | 0.7200 | 1.0552 |
| **3** | 0.8600 | 0.9674 | 0.2100 | 0.8218 | 0.1700 | 1.2420 | 0.1100 | 0.7899 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6300 | 0.9238 | 0.6300 | 1.0312 | 0.3100 | 0.8604 | 0.5100 | 1.0581 |
| **2** | 0.4100 | 1.1541 | 0.3100 | 1.0762 | 0.0500 | 1.2621 | 0.1900 | 0.9248 |
| **3** | 0.0700 | 1.3562 | 0.3400 | 1.1153 | 0.7300 | 0.9472 | 0.2200 | 0.8951 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8800 | 0.9748 | 0.8800 | 1.0080 | 0.8000 | 1.0181 | 0.6900 | 0.9770 |
| **2** | 0.5000 | 0.9100 | 0.3100 | 0.9305 | 0.4500 | 1.0484 | 0.6500 | 1.0210 |
| **3** | 0.8500 | 0.9699 | 0.0800 | 1.2263 | 0.4000 | 1.0743 | 0.8000 | 1.0189 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3600 | 1.1730 | 0.5900 | 0.9074 | 0.4800 | 1.0566 | 0.3800 | 1.0487 |
| **2** | 0.6600 | 0.9323 | 0.9900 | 0.9979 | 0.0600 | 0.8771 | 0.5200 | 0.9683 |
| **3** | 0.8000 | 1.0373 | 0.4600 | 1.1205 | 0.3700 | 1.0755 | 0.4500 | 0.9534 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1500 | 1.2846 | 0.0900 | 1.1566 | 0.8500 | 1.0228 | 0.2400 | 1.2644 |
| **2** | 0.2700 | 1.1908 | 0.4100 | 0.9313 | 0.3300 | 0.8907 | 0.2200 | 1.2964 |
| **3** | 0.3900 | 1.1470 | 0.4500 | 0.9168 | 0.6500 | 0.9549 | 0.3700 | 0.8500 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0600 | 1.3141 | 0.1200 | 1.3032 | 0.9700 | 1.0048 | 0.3100 | 1.0674 |
| **2** | 0.2800 | 0.8389 | 0.3400 | 0.8764 | 0.4000 | 0.9303 | 0.2100 | 1.0612 |
| **3** | 0.6400 | 1.0819 | 0.1900 | 1.1883 | 0.9000 | 0.9876 | 0.4600 | 0.9549 |

Table 159. *t*-Test ACSE/DACSE - File 7 (15 Nodes)

| *t*-Test & DACSE/ACSE Mean Ratio Results - 15 Node (File 7) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0300** | 0.7566 | 0.9900 | 0.9984 | 0.5000 | 1.1317 | 0.3400 | 1.1728 |
| **2** | **0.0300** | 1.3454 | 0.4600 | 0.9103 | 0.3600 | 0.8880 | 0.6300 | 1.0549 |
| **3** | 0.2800 | 0.8715 | 0.0700 | 1.3776 | 0.1400 | 1.2335 | 0.4900 | 1.1194 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5600 | 1.0768 | 0.9100 | 0.9797 | 0.4100 | 0.8491 | 0.6200 | 1.0601 |
| **2** | 0.9700 | 1.0049 | 0.2500 | 0.7719 | 0.8600 | 0.9700 | 0.8800 | 1.0193 |
| **3** | 0.1400 | 1.2225 | 0.7900 | 0.9537 | 0.5900 | 0.8848 | 0.2400 | 1.1746 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3200 | 1.1381 | 0.7000 | 1.0505 | 0.0600 | 1.1958 | 0.8500 | 0.9900 |
| **2** | 0.6500 | 1.0468 | 0.5800 | 1.0593 | 0.9500 | 1.0075 | 0.5600 | 1.0253 |
| **3** | 0.3000 | 1.1942 | 0.0500 | 0.7710 | 0.0800 | 1.1913 | **0.0100** | 0.8259 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8700 | 0.9804 | **0.0100** | 1.5193 | 0.8000 | 1.0136 | 0.5400 | 1.0416 |
| **2** | 0.1600 | 0.8461 | 0.1500 | 1.2668 | 0.8100 | 0.9892 | **0.0000** | 1.2232 |
| **3** | 0.1500 | 1.2164 | 0.5800 | 1.0961 | 0.5900 | 0.9723 | 0.2100 | 0.9075 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3700 | 0.8856 | 0.6000 | 1.0640 | 0.1100 | 1.1461 | 0.3500 | 1.0943 |
| **2** | 0.3700 | 0.8846 | 0.2800 | 0.8991 | 0.6900 | 1.0260 | 0.0500 | 1.1027 |
| **3** | 0.2600 | 1.1744 | 0.7800 | 0.9668 | 0.1400 | 1.1611 | 0.3900 | 0.9265 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6400 | 0.9230 | 0.4800 | 1.0812 | 0.3400 | 1.1254 | 0.1500 | 1.1074 |
| **2** | 0.4600 | 1.1003 | 0.3300 | 1.0941 | 0.3200 | 0.8892 | 0.9400 | 1.0029 |
| **3** | 0.4000 | 0.8682 | 0.5800 | 1.0585 | 0.3100 | 0.8793 | **0.0000** | 1.2676 |

## Table 160. *t*-Test ACSE/DACSE - File 8 (15 Nodes)

| *t*-Test & DACSE/ACSE Mean Ratio Results - 15 Node (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2300 | 1.2470 | **0.0100** | 1.5266 | 0.2500 | 1.2158 | 0.3700 | 1.1507 |
| **2** | 0.9900 | 0.9981 | 0.2000 | 1.2858 | 0.6000 | 0.9161 | 0.7300 | 0.9308 |
| **3** | 0.1200 | 1.3210 | 0.3900 | 1.2243 | 0.2400 | 1.3062 | 0.3300 | 1.2347 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2800 | 1.2020 | 0.2700 | 0.8615 | 0.3800 | 1.1562 | 0.8700 | 0.9914 |
| **2** | 0.6800 | 1.0879 | 0.7000 | 1.0554 | 0.6700 | 0.9409 | 0.3200 | 1.1647 |
| **3** | 0.6400 | 1.0944 | 0.6500 | 1.0770 | 0.7700 | 1.0489 | 0.4600 | 1.1264 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2900 | 1.1877 | 0.1900 | 1.1949 | 0.5400 | 0.9671 | 0.2900 | 1.0273 |
| **2** | 0.4800 | 1.1562 | 0.4800 | 0.8941 | 0.4800 | 1.0423 | 0.8700 | 0.9950 |
| **3** | 0.4800 | 0.8867 | 0.1200 | 0.8310 | 0.6900 | 0.9781 | 0.8300 | 0.9909 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8800 | 1.0262 | 0.9500 | 0.9893 | 0.6500 | 1.0215 | 0.0800 | 0.9486 |
| **2** | 0.8400 | 1.0398 | 0.9100 | 1.0273 | 0.3100 | 0.9564 | 0.7300 | 0.9804 |
| **3** | 0.9500 | 1.0105 | 0.8700 | 0.9735 | 0.6400 | 1.0265 | 0.9400 | 1.0035 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7900 | 1.0499 | 0.5800 | 0.9353 | 0.1000 | 1.1454 | 0.6600 | 0.9855 |
| **2** | 0.6700 | 0.9221 | 0.8500 | 1.0334 | 0.1600 | 1.1251 | 0.4600 | 1.0109 |
| **3** | 0.5700 | 1.0971 | 0.3000 | 1.1921 | 0.7600 | 0.9719 | **0.0000** | 1.0644 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0500 | 1.4028 | 0.5100 | 1.0891 | 0.4700 | 1.0577 | **0.0400** | 1.0468 |
| **2** | 0.1400 | 1.2918 | 0.5000 | 1.1093 | **0.0100** | 1.1514 | 0.6000 | 1.0017 |
| **3** | **0.0400** | 1.4741 | 0.4000 | 1.1309 | 0.2500 | 0.8754 | 0.7800 | 1.0036 |

**Table 161. *t*-Test ACSE/DACSE - File 9 (15 Nodes)**

| *t*-Test & DACSE/ACSE Mean Ratio Results - 15 Node (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1800 | 0.8401 | 0.8000 | 0.9640 | 0.8900 | 0.8306 | **0.0200** | 1.2361 |
| **2** | 0.8200 | 1.2374 | 0.3200 | 1.0165 | 0.9300 | 1.1272 | 0.5500 | 1.0552 |
| **3** | 0.7900 | 0.9674 | 0.0600 | 0.8218 | 0.3600 | 1.2420 | 0.1600 | 0.7899 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3300 | 0.9238 | 0.6100 | 1.0312 | 0.7100 | 0.8604 | 0.1600 | 1.0581 |
| **2** | 0.5900 | 1.1541 | 0.8400 | 1.0762 | 0.1100 | 1.2621 | 0.0500 | 0.9248 |
| **3** | 0.1000 | 1.3562 | 0.6500 | 1.1153 | 0.8000 | 0.9472 | 0.4800 | 0.8951 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2400 | 0.9748 | 0.6800 | 1.0080 | 0.7500 | 1.0181 | 0.7600 | 0.9770 |
| **2** | 0.3500 | 0.9100 | 0.5000 | 0.9305 | 0.2500 | 1.0484 | 0.4800 | 1.0210 |
| **3** | 0.7600 | 0.9699 | 0.1600 | 1.2263 | 0.3600 | 1.0743 | 0.0600 | 1.0189 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0700 | 1.1730 | 0.1700 | 0.9074 | 0.9100 | 1.0566 | 0.0500 | 1.0487 |
| **2** | 0.0800 | 0.9323 | 0.1200 | 0.9979 | 0.6900 | 0.8771 | 0.1200 | 0.9683 |
| **3** | 0.6200 | 1.0373 | **0.0000** | 1.1205 | 0.9200 | 1.0755 | **0.0300** | 0.9534 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3300 | 1.2846 | 0.6700 | 1.1566 | 0.1900 | 1.0228 | **0.0100** | 1.2644 |
| **2** | 0.6300 | 1.1908 | 0.3200 | 0.9313 | 0.2200 | 0.8907 | 0.2700 | 1.2964 |
| **3** | 0.1800 | 1.1470 | 0.9900 | 0.9168 | 0.2400 | 0.9549 | 0.7700 | 0.8500 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9100 | 1.3141 | 0.8300 | 1.3032 | 0.2800 | 1.0048 | 0.0600 | 1.0674 |
| **2** | 0.4400 | 0.8389 | 0.0800 | 0.8764 | 0.4000 | 0.9303 | 0.7200 | 1.0612 |
| **3** | 0.9500 | 1.0819 | 0.8600 | 1.1883 | 0.1100 | 0.9876 | 0.4000 | 0.9549 |

**Table 162. *t*-Test ACSE/DACSE - File 10 (15 Nodes)**

| ***t*-Test & DACSE/ACSE Mean Ratio Results - 15 Node (File 10)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8100 | 0.8401 | 0.8000 | 0.9640 | 0.5900 | 0.8306 | 0.4700 | 1.2361 |
| **2** | 0.3800 | 1.2374 | 0.4000 | 1.0165 | 0.8700 | 1.1272 | 0.5100 | 1.0552 |
| **3** | 0.0700 | 0.9674 | 0.7500 | 0.8218 | 0.5700 | 1.2420 | 0.5400 | 0.7899 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1200 | 0.9238 | **0.0100** | 1.0312 | 0.7600 | 0.8604 | 0.7900 | 1.0581 |
| **2** | 0.7700 | 1.1541 | 0.2300 | 1.0762 | 0.4200 | 1.2621 | 0.7800 | 0.9248 |
| **3** | 0.1800 | 1.3562 | 0.9300 | 1.1153 | 0.4200 | 0.9472 | 0.5600 | 0.8951 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3400 | 0.9748 | 0.1500 | 1.0080 | 0.6300 | 1.0181 | 0.1100 | 0.9770 |
| **2** | 0.0800 | 0.9100 | 0.6900 | 0.9305 | 0.9500 | 1.0484 | 0.8300 | 1.0210 |
| **3** | 0.4500 | 0.9699 | 0.2000 | 1.2263 | 0.6400 | 1.0743 | 0.7200 | 1.0189 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5900 | 1.1730 | 0.9100 | 0.9074 | 0.2700 | 1.0566 | 0.8700 | 1.0487 |
| **2** | 0.9600 | 0.9323 | 0.7000 | 0.9979 | 0.5000 | 0.8771 | 0.1100 | 0.9683 |
| **3** | 0.2900 | 1.0373 | 0.0900 | 1.1205 | 0.5600 | 1.0755 | 0.9000 | 0.9534 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8500 | 1.2846 | 0.5400 | 1.1566 | 0.9000 | 1.0228 | 0.4200 | 1.2644 |
| **2** | 0.0600 | 1.1908 | 0.1500 | 0.9313 | 0.4200 | 0.8907 | 0.9300 | 1.2964 |
| **3** | 0.4700 | 1.1470 | 0.5000 | 0.9168 | 0.3000 | 0.9549 | 0.9900 | 0.8500 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2000 | 1.3141 | **0.0400** | 1.3032 | 0.3300 | 1.0048 | 0.4900 | 1.0674 |
| **2** | 0.3400 | 0.8389 | 0.0900 | 0.8764 | 0.3100 | 0.9303 | 0.6600 | 1.0612 |
| **3** | 0.6700 | 1.0819 | 0.2500 | 1.1883 | 0.2300 | 0.9876 | **0.0300** | 0.9549 |

## Appendix J: *t*-Tests ACSS/DACSS (15 Nodes)

### Table 163. *t*-Tests ACSS/DACSS - File 1 (15 Nodes)

| *t*-Test & DACSS/ACSS Mean Ratio Results - 15 Node (File 1) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1600 | 0.9881 | **0.0400** | 1.0153 | 0.7600 | 1.0019 | **0.0300** | 1.0232 |
| **2** | **0.0000** | 1.0226 | 0.3200 | 0.9920 | 0.2300 | 0.9911 | 0.5500 | 1.0076 |
| **3** | 0.4100 | 1.0094 | 0.1700 | 1.0124 | 0.1900 | 1.0112 | 0.3800 | 1.0051 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2700 | 0.9919 | 0.0600 | 1.0226 | 0.0600 | 1.0307 | 0.4000 | 1.0214 |
| **2** | 0.2300 | 1.0093 | 0.6200 | 0.9950 | 0.6200 | 1.0076 | 0.3300 | 1.0098 |
| **3** | 0.4400 | 0.9905 | 0.2500 | 0.9809 | 0.2100 | 1.0095 | 0.7500 | 0.9982 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0000** | 1.0187 | 0.4400 | 0.9969 | 0.5400 | 1.0050 | 0.3300 | 1.0052 |
| **2** | 0.6200 | 0.9956 | 0.7700 | 1.0022 | 0.2100 | 1.0126 | 0.2800 | 0.9903 |
| **3** | 0.0700 | 1.0252 | 0.1100 | 1.0176 | 0.2500 | 1.0132 | 0.0600 | 1.0197 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4000 | 1.0066 | 0.3800 | 0.9921 | 0.7700 | 0.9993 | 0.3500 | 0.9039 |
| **2** | 0.6200 | 0.9949 | 0.6500 | 0.9966 | 0.9900 | 1.0000 | 0.4500 | 0.8743 |
| **3** | **0.0300** | 1.0165 | 0.4600 | 1.0066 | **0.0400** | 0.9921 | 0.6500 | 0.9929 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0900 | 1.0144 | **0.0300** | 1.0020 | 0.8900 | 1.0031 | 0.7800 | 0.9989 |
| **2** | 0.0500 | 0.9868 | 0.1600 | 1.0193 | 0.9800 | 1.0003 | **0.0400** | 0.9916 |
| **3** | 0.3900 | 1.0100 | 0.4300 | 1.0006 | 0.4000 | 1.0239 | 0.8800 | 0.9993 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0900 | 0.9899 | 0.1800 | 1.0178 | 0.9300 | 1.0013 | **0.0100** | 0.8390 |
| **2** | 0.3900 | 0.9783 | 0.6500 | 0.9911 | **0.0300** | 0.9742 | 0.9500 | 1.0103 |
| **3** | 0.0800 | 1.0354 | 0.1400 | 1.0150 | 0.6400 | 0.9973 | 0.6400 | 0.9329 |

**Table 164. *t*-Test ACSS/DACSS - File 2 (15 Nodes)**

| ***t*-Test & DACSS/ACSS Mean Ratio Results - 15 Node (File 2)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0800 | 1.0334 | 0.7200 | 0.9932 | **0.0300** | 1.0462 | **0.0300** | 0.9562 |
| **2** | 0.2300 | 1.0272 | 0.4900 | 0.9865 | 0.8200 | 0.9958 | 0.0900 | 0.9713 |
| **3** | 0.7000 | 1.0074 | **0.0400** | 0.9216 | 0.1200 | 0.9594 | 0.7300 | 1.0094 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7500 | 1.0050 | 0.4700 | 0.9918 | 0.4300 | 0.9911 | 0.1700 | 1.0090 |
| **2** | 0.7200 | 0.9939 | 0.9700 | 0.9995 | 0.2900 | 0.9702 | 0.2800 | 1.0051 |
| **3** | 0.7000 | 1.0074 | 0.0700 | 0.9850 | 0.4100 | 1.0117 | 0.6000 | 0.9949 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3100 | 1.0188 | **0.0000** | 1.0520 | 0.9600 | 1.0003 | 0.3900 | 1.0162 |
| **2** | 0.2100 | 0.9790 | 0.4400 | 1.0120 | **0.0200** | 1.0229 | 0.2300 | 1.0157 |
| **3** | 0.2900 | 0.9753 | **0.0400** | 1.0307 | 0.9700 | 0.9998 | 0.9800 | 0.9996 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1100 | 0.9826 | **0.0400** | 1.0011 | **0.0200** | 1.0009 | 0.4300 | 0.9995 |
| **2** | 0.7400 | 0.9930 | **0.0100** | 1.0008 | 0.4200 | 0.9997 | 0.2600 | 0.9989 |
| **3** | 0.6300 | 0.9811 | 0.0500 | 0.9991 | 0.4700 | 1.0002 | 0.8300 | 1.0002 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5500 | 0.9884 | 0.8600 | 0.9979 | 0.7800 | 1.0734 | 0.3600 | 0.9536 |
| **2** | 0.5700 | 1.0108 | **0.0200** | 1.0256 | 0.1200 | 0.8441 | 0.1400 | 1.0855 |
| **3** | 0.1500 | 0.9766 | **0.0100** | 1.0428 | 0.7400 | 1.0917 | 0.7900 | 0.9876 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7900 | 1.0113 | 0.1900 | 1.0208 | 0.4700 | 1.0075 | 0.0500 | 1.0389 |
| **2** | **0.0100** | 1.0316 | 0.7200 | 1.0076 | 0.3200 | 0.9865 | 0.2000 | 0.9835 |
| **3** | 0.1200 | 0.9740 | **0.0000** | 0.9611 | 0.1100 | 1.0185 | 0.1200 | 0.9578 |

**Table 165. *t*-Test ACSS/DACSS - File 3 (15 Nodes)**

| *t*-Test & DACSS/ACSS Mean Ratio Results - 15 Node (File 3) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4500 | 1.0085 | 0.3600 | 1.0050 | 0.7000 | 0.9965 | **0.0000** | 1.0249 |
| **2** | **0.0000** | 1.0263 | **0.0100** | 1.0135 | **0.0200** | 1.0157 | **0.0400** | 1.0288 |
| **3** | 1.0000 | 1.0000 | 0.3100 | 1.0079 | 0.8300 | 0.9981 | 0.7900 | 1.0020 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2200 | 0.9910 | 0.2500 | 1.1056 | 0.3100 | 0.9778 | 0.9000 | 1.0296 |
| **2** | 0.4200 | 0.9913 | 0.5700 | 0.9076 | 0.2500 | 1.0230 | 0.9200 | 0.9746 |
| **3** | 0.9000 | 1.0018 | 0.3900 | 1.0063 | 0.2700 | 0.9840 | 0.4100 | 0.8336 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9000 | 0.9990 | 0.1700 | 1.0144 | 0.4800 | 0.8523 | 0.9400 | 1.0122 |
| **2** | 0.0600 | 1.0114 | 0.5900 | 1.0053 | 0.5900 | 0.8836 | 0.5600 | 0.9444 |
| **3** | 0.8900 | 0.9990 | 0.2000 | 1.0154 | 0.3400 | 1.1864 | 0.8300 | 1.0266 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0300** | 0.9615 | 0.9400 | 0.9995 | **0.0300** | 0.9599 | 0.6000 | 1.1402 |
| **2** | 0.1500 | 1.0096 | 0.2100 | 0.9864 | 0.9200 | 1.0026 | 0.5200 | 0.8447 |
| **3** | 0.2100 | 1.0108 | 0.6800 | 1.0063 | 0.1400 | 1.0299 | 0.3600 | 1.1732 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7800 | 1.0013 | 0.2500 | 0.8582 | 0.9600 | 1.0126 | 0.2000 | 1.0267 |
| **2** | 0.1900 | 1.0067 | 0.2500 | 1.1229 | 0.5000 | 0.9383 | 0.8800 | 0.9939 |
| **3** | **0.0200** | 1.0189 | 0.0800 | 0.9662 | 0.9700 | 1.0052 | **0.0200** | 0.9359 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7600 | 1.0009 | 0.0700 | 1.0201 | 0.8300 | 1.0001 | 0.1000 | 1.0585 |
| **2** | **0.0000** | 1.0178 | 0.2500 | 0.9910 | 0.2500 | 1.0005 | 0.3500 | 0.9445 |
| **3** | **0.0100** | 1.0131 | 0.5000 | 0.9902 | 0.4900 | 1.0003 | 0.5800 | 0.9834 |

**Table 166. *t*-Test ACSS/DACSS - File 4 (15 Nodes)**

| ***t*-Test & DACSS/ACSS Mean Ratio Results - 15 Node (File 4)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0700 | 0.9355 | **0.0100** | 0.9979 | 0.3800 | 1.0087 | **0.0000** | 1.0034 |
| **2** | 0.3500 | 0.9995 | **0.0100** | 1.0015 | 0.1400 | 0.9987 | 0.7500 | 1.0002 |
| **3** | 0.6700 | 0.9996 | **0.0200** | 0.9985 | 0.0600 | 0.9987 | 0.4900 | 1.0005 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1800 | 1.0014 | 0.4700 | 1.0003 | 0.1600 | 1.0186 | 0.2500 | 0.9793 |
| **2** | 0.2100 | 0.9932 | 0.3300 | 0.9993 | 0.2400 | 1.0005 | 0.1500 | 0.9780 |
| **3** | 0.5600 | 1.0002 | **0.0000** | 0.9982 | **0.0300** | 1.0014 | **0.0400** | 1.0224 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5800 | 0.9996 | 0.6300 | 0.9997 | 0.6700 | 1.0980 | 0.3100 | 1.0149 |
| **2** | 0.1300 | 0.9899 | 0.2200 | 0.9993 | 0.9100 | 1.0158 | 0.2700 | 0.8512 |
| **3** | 0.7300 | 1.0002 | **0.0000** | 1.0015 | 0.5100 | 1.1753 | **0.0300** | 1.0277 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5000 | 0.9995 | 0.1500 | 0.9869 | 0.1400 | 1.0002 | 0.2000 | 0.9901 |
| **2** | 0.4400 | 0.9993 | 0.1300 | 1.0112 | 0.3900 | 1.0001 | 0.0500 | 0.9906 |
| **3** | **0.0000** | 1.0033 | 0.2500 | 1.0045 | **0.0000** | 1.0006 | **0.0300** | 0.9807 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4500 | 0.9993 | 0.4600 | 1.1289 | 0.0900 | 1.0177 | **0.0200** | 1.1159 |
| **2** | 0.0700 | 0.9984 | 0.1900 | 1.2246 | 0.1700 | 1.2102 | 0.6300 | 1.0173 |
| **3** | 0.6800 | 0.9998 | 0.1400 | 1.0151 | **0.0000** | 1.0348 | 0.6400 | 0.9801 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5800 | 1.0005 | **0.0300** | 1.0316 | 0.6800 | 0.9895 | 0.1200 | 0.9793 |
| **2** | 0.2800 | 1.0011 | **0.0100** | 1.0207 | **0.0400** | 1.0432 | 0.2000 | 1.0136 |
| **3** | 0.4700 | 1.0005 | 0.5700 | 0.9921 | 0.5500 | 1.0120 | 0.1200 | 1.0278 |

## Table 167. *t*-Test ACSS/DACSS - File 5 (15 Nodes)

| *t*-Test & DACSS/ACSS Mean Ratio Results - 15 Node (File 5) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0900 | 0.9870 | 0.3700 | 1.0055 | **0.0400** | 1.0207 | 0.6000 | 0.9967 |
| **2** | 0.0800 | 1.0136 | **0.0200** | 1.0089 | 0.7100 | 1.0024 | **0.0000** | 1.0302 |
| **3** | **0.0000** | 1.0242 | **0.0300** | 1.0157 | **0.0400** | 1.0224 | **0.0300** | 1.0205 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0900 | 1.0178 | 0.8700 | 0.9989 | **0.0300** | 1.0150 | 0.1800 | 0.9913 |
| **2** | 0.3400 | 1.0084 | 0.1600 | 1.0165 | 0.6400 | 0.9952 | 0.7900 | 0.9985 |
| **3** | **0.0100** | 1.0210 | 0.5900 | 1.0059 | 0.1400 | 1.0109 | 0.0700 | 1.0216 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4900 | 1.0068 | 0.3200 | 1.1191 | 0.6600 | 0.9972 | 0.3200 | 0.9753 |
| **2** | 0.6900 | 0.9966 | 0.6400 | 1.0096 | 0.2600 | 1.0342 | 0.1100 | 1.0344 |
| **3** | **0.0100** | 1.0310 | 0.3400 | 0.9810 | 0.2300 | 0.9825 | 0.4500 | 1.0126 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3200 | 0.9949 | **0.0300** | 1.0174 | 0.5800 | 0.8505 | 0.9100 | 1.0381 |
| **2** | **0.0200** | 1.0138 | **0.0000** | 1.0272 | 0.9100 | 1.0333 | 0.0600 | 0.5575 |
| **3** | 0.0500 | 1.0199 | 0.8300 | 1.0022 | 0.4500 | 0.8510 | 0.1900 | 0.6599 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0300** | 1.0170 | 0.3200 | 0.9686 | 0.2600 | 1.0062 | 0.3000 | 0.9537 |
| **2** | **0.0300** | 1.0183 | 0.8300 | 1.0028 | **0.0100** | 1.0168 | **0.0100** | 1.1592 |
| **3** | 0.3100 | 0.9914 | 0.2800 | 1.0089 | 0.1700 | 0.9861 | 0.2400 | 1.0724 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9200 | 1.0008 | 0.6100 | 1.1520 | 0.0500 | 1.0302 | 0.5100 | 0.9892 |
| **2** | **0.0100** | 1.0182 | 0.3600 | 0.7939 | 0.1600 | 0.9700 | **0.0100** | 0.9789 |
| **3** | 0.2400 | 1.0149 | 0.3300 | 0.9848 | 0.1500 | 1.0125 | 0.5800 | 0.9899 |

## Table 168. *t*-Test ACSS/DACSS - File 6 (15 Nodes)

| *t*-Test & DACSS/ACSS Mean Ratio Results - 15 Node (File 6) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7400 | 0.9964 | **0.0100** | 1.0429 | **0.0100** | 1.0232 | **0.0100** | 1.0322 |
| **2** | **0.0000** | 1.0337 | **0.0100** | 1.0195 | 0.1200 | 1.0152 | 0.3500 | 1.0050 |
| **3** | 0.1200 | 0.9806 | **0.0200** | 1.1574 | 0.4100 | 0.9937 | 0.2500 | 1.0214 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0900 | 0.9825 | 0.1400 | 1.2962 | 0.6900 | 0.9968 | **0.0000** | 1.4135 |
| **2** | 0.8000 | 0.9978 | 0.7100 | 1.0689 | 0.5200 | 0.9938 | 0.4200 | 1.1538 |
| **3** | 0.2200 | 1.0067 | 0.8800 | 0.9978 | **0.0100** | 1.0336 | 0.2900 | 0.8839 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8600 | 1.0015 | 0.1600 | 1.0255 | 0.5100 | 1.1073 | 0.9700 | 1.0062 |
| **2** | 0.8200 | 1.0033 | 0.6900 | 0.9944 | 0.6300 | 1.1328 | 0.7300 | 1.0784 |
| **3** | 0.8500 | 1.0020 | 0.1200 | 1.0244 | 0.2200 | 0.7696 | 0.1100 | 0.7291 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4200 | 1.0105 | 0.1000 | 1.0242 | 0.5800 | 1.1319 | 0.0800 | 0.7272 |
| **2** | 0.1500 | 0.9883 | 0.2700 | 0.9863 | 0.6200 | 1.1361 | **0.0200** | 1.2311 |
| **3** | 0.8200 | 1.0027 | 0.3500 | 1.0111 | 0.5600 | 1.1532 | 0.7400 | 0.9421 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.5400 | 0.9963 | 0.3500 | 0.8608 | 0.4800 | 0.9917 | 0.2800 | 0.9873 |
| **2** | **0.0300** | 1.0191 | 0.1300 | 1.3061 | 0.4800 | 0.9865 | 0.9500 | 0.9993 |
| **3** | 0.8800 | 0.9987 | 0.1500 | 1.2375 | 0.1700 | 0.9703 | 0.3300 | 0.9878 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0900 | 1.0195 | 0.2500 | 0.8322 | **0.0300** | 1.0340 | 0.5900 | 1.0036 |
| **2** | **0.0000** | 1.0546 | 0.1400 | 1.2553 | 0.2000 | 0.9820 | **0.0400** | 1.0000 |
| **3** | **0.0000** | 1.0412 | 0.2000 | 1.2071 | 0.6600 | 0.9937 | **0.0100** | 1.1050 |

**Table 169. *t*-Test ACSS/DACSS - File 7 (15 Nodes)**

| *t*-Test & DACSS/ACSS Mean Ratio Results - 15 Node (File 7) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.2100 | 1.0200 | **0.0000** | 1.0268 | 0.3700 | 1.0114 | 0.0500 | 0.9742 |
| **2** | **0.0400** | 0.9842 | 0.7700 | 1.0035 | **0.0100** | 0.9476 | 0.0600 | 1.0123 |
| **3** | **0.0000** | 1.0638 | 0.0500 | 1.0182 | **0.0400** | 1.0225 | **0.0400** | 0.9745 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0100** | 1.0292 | **0.0000** | 1.0474 | 0.1200 | 0.9756 | 0.2600 | 1.1471 |
| **2** | 0.3400 | 1.0096 | 0.3600 | 1.0172 | 0.6600 | 0.9906 | 0.7800 | 0.9954 |
| **3** | **0.0100** | 1.0671 | 0.3000 | 0.9830 | 0.1500 | 1.0156 | 0.0500 | 0.9667 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4700 | 0.9912 | 0.3500 | 0.9781 | 0.4200 | 1.1496 | 0.0700 | 0.7833 |
| **2** | **0.0000** | 0.9176 | 0.4200 | 0.9815 | 0.4300 | 1.1501 | 0.4700 | 0.8980 |
| **3** | 0.2500 | 1.0190 | 0.2100 | 0.9815 | 0.3800 | 1.2439 | 0.9700 | 0.9917 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.9100 | 0.9988 | 0.2700 | 1.0125 | 0.2300 | 1.2041 | 0.2600 | 1.3404 |
| **2** | **0.0300** | 0.9429 | 0.5300 | 0.9963 | 0.1000 | 1.2660 | 0.4200 | 1.2097 |
| **3** | 0.0900 | 1.1208 | 0.3400 | 1.0079 | 0.2300 | 0.7258 | 0.5500 | 0.8611 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1400 | 0.9440 | 0.6800 | 1.0035 | 0.2000 | 0.8250 | 0.5400 | 1.1018 |
| **2** | 0.2000 | 1.0199 | 0.6500 | 0.9952 | 0.3800 | 0.8186 | 0.1700 | 1.3378 |
| **3** | 0.1000 | 1.0428 | **0.0200** | 0.9836 | 0.3700 | 0.9767 | 0.6100 | 1.1127 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1200 | 0.9770 | 0.0500 | 1.3064 | 0.3100 | 1.0143 | 0.6700 | 0.9139 |
| **2** | 0.1200 | 0.9695 | 0.9400 | 0.9880 | **0.0300** | 0.9641 | 0.3600 | 1.2708 |
| **3** | 0.6000 | 1.0077 | 0.1200 | 0.7140 | 0.0500 | 0.9678 | 0.3100 | 0.8763 |

**Table 170. *t*-Test ACSS/DACSS - File 8 (15 Nodes)**

| *t*-Test & DACSS/ACSS Mean Ratio Results - 15 Node (File 8) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1900 | 0.9846 | **0.0300** | 0.8960 | 0.4200 | 0.9919 | 0.0500 | 1.0138 |
| **2** | **0.0000** | 1.0169 | 0.8400 | 0.9988 | 0.7100 | 1.0038 | **0.0100** | 1.0314 |
| **3** | **0.0000** | 1.0508 | 0.5500 | 0.9920 | **0.0000** | 1.0434 | **0.0100** | 0.9415 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1500 | 1.0841 | 0.1600 | 1.0114 | **0.0200** | 0.9699 | **0.0000** | 1.0488 |
| **2** | 0.1200 | 1.0265 | 0.7600 | 1.0028 | **0.0200** | 1.0346 | 0.1400 | 1.0210 |
| **3** | 0.6000 | 0.9944 | 0.6800 | 0.9969 | 0.0500 | 0.9825 | 0.0800 | 1.0260 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1700 | 1.0541 | 0.7200 | 0.9093 | 0.2300 | 0.9992 | 0.7200 | 1.0089 |
| **2** | **0.0300** | 1.0180 | 0.3800 | 1.1446 | 0.9200 | 1.0001 | **0.0000** | 1.0582 |
| **3** | 0.3700 | 0.9417 | 0.1900 | 0.7930 | 0.0700 | 1.0008 | **0.0300** | 0.9690 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.1100 | 0.9090 | 0.6200 | 1.0036 | 0.0700 | 0.9983 | 0.8800 | 1.0038 |
| **2** | 0.0700 | 1.1219 | 0.8900 | 0.9989 | 0.1900 | 0.9833 | 0.3000 | 0.9788 |
| **3** | 0.2400 | 0.9606 | **0.0300** | 1.0397 | 0.4600 | 1.0052 | 0.1000 | 0.9805 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.7700 | 0.9973 | **0.0200** | 1.0177 | 0.5200 | 0.8671 | 0.7100 | 1.0000 |
| **2** | 0.4300 | 1.0155 | 0.4100 | 0.9952 | 0.6000 | 1.0941 | 0.9100 | 1.0000 |
| **3** | 0.1000 | 1.0461 | 0.2800 | 1.0115 | 0.9700 | 1.0082 | 0.8000 | 1.0000 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4400 | 1.0096 | 0.1200 | 1.1603 | 0.3800 | 1.1336 | 0.4900 | 1.0328 |
| **2** | 0.7500 | 0.9972 | 0.2900 | 0.9889 | 0.5200 | 1.1588 | 0.2700 | 0.9439 |
| **3** | 0.4600 | 1.0075 | 0.2300 | 0.7860 | 0.5400 | 0.8787 | **0.0100** | 1.1197 |

## Table 171. *t*-Test ACSS/DACSS - File 9 (15 Nodes)

| *t*-Test & DACSS/ACSS Mean Ratio Results - 15 Node (File 9) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0000** | 1.0190 | **0.0000** | 1.0129 | 0.4500 | 0.9964 | 0.0600 | 1.0258 |
| **2** | 0.0500 | 0.9891 | 0.1600 | 0.9919 | **0.0300** | 1.0130 | **0.0000** | 1.0136 |
| **3** | 0.7700 | 1.0015 | **0.0100** | 1.0250 | **0.0200** | 1.0227 | 0.8600 | 1.0008 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0700 | 1.0122 | **0.0200** | 1.0547 | 0.3200 | 1.5575 | 0.1400 | 1.2334 |
| **2** | 0.9800 | 0.9998 | 0.1900 | 1.0153 | 0.6000 | 0.8690 | 0.1500 | 0.7518 |
| **3** | 0.3700 | 0.9941 | 0.7400 | 0.9951 | 0.2300 | 1.3128 | 0.3800 | 1.1774 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0100** | 1.0709 | 0.7700 | 0.9987 | **0.0200** | 1.0150 | 0.8900 | 0.9972 |
| **2** | 0.7700 | 0.9984 | 0.0700 | 0.9909 | 0.1200 | 1.0075 | 0.1100 | 0.8236 |
| **3** | **0.0000** | 1.0175 | 0.4200 | 0.9929 | 0.4600 | 1.0057 | 0.1900 | 1.0186 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0500 | 1.0335 | 0.1200 | 0.7828 | 0.2200 | 0.7437 | 0.2000 | 0.6974 |
| **2** | 0.6100 | 1.0042 | **0.0300** | 1.0398 | 0.1700 | 1.2428 | 0.2800 | 1.2622 |
| **3** | 0.3000 | 1.0085 | **0.0200** | 1.0524 | 0.1700 | 0.6672 | 0.9500 | 0.9895 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.6200 | 0.9969 | **0.0300** | 1.0127 | **0.0200** | 1.0077 | 0.5800 | 1.0371 |
| **2** | 0.0700 | 1.0215 | **0.0100** | 1.0122 | 0.8800 | 0.9994 | 0.1600 | 1.1317 |
| **3** | 0.0600 | 1.0188 | **0.0300** | 1.0195 | 0.0500 | 0.9885 | 0.1300 | 1.1196 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0000** | 1.0202 | 0.3300 | 0.9896 | **0.0100** | 0.7515 | **0.0100** | 1.0913 |
| **2** | **0.0200** | 1.0085 | 0.0900 | 1.0163 | 0.4200 | 1.0620 | 0.1500 | 0.9645 |
| **3** | 0.3800 | 0.9846 | 0.1100 | 0.9720 | 0.7600 | 0.9653 | 0.4800 | 1.0091 |

**Table 172. *t*-Test ACSS/DACSS - File 10 (15 Nodes)**

| ***t*-Test & DACSS/ACSS Mean Ratio Results - 15 Node (File 10)** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.4700 | 1.0022 | 0.0900 | 1.0145 | **0.0000** | 1.0148 | 0.3200 | 1.0051 |
| **2** | **0.0000** | 1.0107 | **0.0000** | 1.0115 | **0.0000** | 1.0196 | 0.1700 | 0.9838 |
| **3** | **0.0000** | 1.0269 | **0.0000** | 1.0242 | 0.5400 | 0.9869 | 0.8700 | 0.9993 |
| **10%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.3800 | 0.9920 | 0.0600 | 0.9654 | **0.0100** | 1.0252 | 0.3700 | 0.9629 |
| **2** | 0.1900 | 1.0087 | 0.0700 | 1.0190 | 0.1600 | 0.9713 | 0.5300 | 1.0151 |
| **3** | 0.0700 | 1.0148 | 0.5400 | 1.0065 | 0.1400 | 1.0116 | 0.2300 | 1.0259 |
| **20%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.0600 | 1.0091 | 0.5500 | 1.0074 | 0.1500 | 1.0089 | **0.0000** | 1.0158 |
| **2** | **0.0100** | 1.0154 | 0.1100 | 0.9720 | 0.0600 | 1.0113 | 0.3500 | 1.0032 |
| **3** | **0.0000** | 0.9692 | 0.4100 | 1.0078 | 0.1400 | 0.9865 | 0.0900 | 1.0183 |
| **30%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | 0.8900 | 1.0008 | 0.9700 | 0.9924 | **0.0100** | 0.7225 | 0.7900 | 1.0079 |
| **2** | 0.3800 | 1.0058 | 0.1500 | 0.8062 | 0.6900 | 0.9313 | 0.8100 | 0.9886 |
| **3** | 0.8700 | 0.9990 | 0.6500 | 0.8919 | 0.7300 | 0.9263 | 0.3800 | 0.9670 |
| **40%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0100** | 1.0201 | 0.8100 | 1.0002 | 0.9700 | 0.9971 | 0.5000 | 0.9657 |
| **2** | 0.0800 | 0.9893 | **0.0100** | 1.0016 | 0.0500 | 0.8612 | **0.0400** | 0.8666 |
| **3** | **0.0000** | 1.0276 | 0.1800 | 1.0007 | 0.6200 | 0.9650 | 0.1800 | 0.9072 |
| **50%** | **Iterations 1:25** | | **Iterations 26:50** | | **Iterations 51:75** | | **Iterations 76:100** | |
| **METHOD** | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | | **p-Value/Ratio** | |
| **1** | **0.0000** | 1.0312 | 0.4800 | 1.0154 | 0.8300 | 0.9984 | 0.8500 | 0.9946 |
| **2** | 0.6900 | 1.0021 | 0.4700 | 0.7876 | 0.1800 | 0.8101 | 0.2400 | 0.8461 |
| **3** | 0.4900 | 1.0065 | 0.1800 | 1.0242 | 0.4200 | 1.0068 | 0.8000 | 1.0107 |

# Appendix K: Timing Results

**Table 173. Timing Results - ACSE (10 Nodes)**

| Timing Results - ACSE (10 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 44.75 | 128.09 | 107.53 | 55.00 | 76.32 | 85.83 | 65.88 | 64.90 | 51.65 | 81.10 |
| **ACSE-2** | 44.06 | 44.02 | 104.04 | 53.56 | 75.58 | 83.02 | 64.24 | 63.07 | 50.48 | 79.17 |
| **ACSE-3** | 43.98 | 123.96 | 103.85 | 53.54 | 75.29 | 83.18 | 64.28 | 63.03 | 50.50 | 79.32 |
| **10%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 41.35 | 94.68 | 84.10 | 38.84 | 54.60 | 77.92 | 50.76 | 58.42 | 45.32 | 48.53 |
| **ACSE-2** | 41.60 | 41.51 | 86.70 | 39.61 | 54.86 | 80.15 | 51.81 | 59.94 | 46.01 | 49.48 |
| **ACSE-3** | 41.30 | 94.83 | 84.14 | 38.78 | 54.47 | 78.13 | 50.75 | 58.38 | 45.33 | 48.55 |
| **20%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 26.27 | 71.49 | 58.15 | 33.02 | 39.31 | 50.96 | 38.15 | 38.16 | 27.07 | 49.21 |
| **ACSE-2** | 26.09 | 26.08 | 56.62 | 32.46 | 39.00 | 49.72 | 37.37 | 37.28 | 26.57 | 48.26 |
| **ACSE-3** | 25.79 | 69.79 | 56.74 | 32.27 | 38.73 | 49.85 | 37.44 | 37.39 | 26.58 | 48.33 |
| **30%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 20.55 | 40.21 | 35.72 | 24.21 | 24.69 | 31.45 | 28.68 | 26.85 | 24.53 | 34.65 |
| **ACSE-2** | 20.64 | 20.61 | 35.05 | 24.31 | 24.57 | 31.85 | 28.64 | 26.75 | 24.58 | 34.58 |
| **ACSE-3** | 20.58 | 39.98 | 35.04 | 24.38 | 24.66 | 32.17 | 28.80 | 26.82 | 24.62 | 34.96 |
| **40%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 24.05 | 54.22 | 37.60 | 23.81 | 31.84 | 33.55 | 29.71 | 25.50 | 28.62 | 35.01 |
| **ACSE-2** | 24.06 | 24.03 | 37.96 | 23.89 | 31.95 | 33.71 | 29.91 | 25.82 | 29.12 | 35.13 |
| **ACSE-3** | 24.07 | 53.99 | 37.65 | 23.78 | 32.06 | 33.60 | 30.37 | 26.68 | 31.03 | 35.15 |
| **50%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 23.76 | 42.68 | 39.82 | 29.99 | 28.82 | 30.21 | 26.36 | 23.82 | 22.08 | 33.19 |
| **ACSE-2** | 23.80 | 23.81 | 39.83 | 29.95 | 28.83 | 30.22 | 26.44 | 23.66 | 23.20 | 33.13 |
| **ACSE-3** | 23.85 | 42.72 | 39.84 | 30.00 | 28.96 | 30.28 | 26.43 | 23.84 | 22.14 | 33.30 |

**Table 174. Timing Results - ACSS (10 Nodes)**

| Timing Results - ACSS (10 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 30.22 | 86.77 | 70.84 | 36.50 | 55.70 | 58.58 | 65.54 | 42.54 | 35.08 | 55.60 |
| **ACSS-2** | 31.03 | 85.72 | 73.57 | 36.35 | 54.99 | 56.37 | 63.67 | 43.80 | 35.75 | 57.01 |
| **ACSS-3** | 29.75 | 89.28 | 71.46 | 35.63 | 54.63 | 55.93 | 64.36 | 43.24 | 34.83 | 56.02 |
| **10%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 29.77 | 69.02 | 58.03 | 26.93 | 41.83 | 54.07 | 55.34 | 41.71 | 30.38 | 37.22 |
| **ACSS-2** | 29.67 | 69.92 | 59.59 | 26.85 | 41.07 | 54.49 | 57.45 | 41.08 | 31.25 | 37.61 |
| **ACSS-3** | 28.92 | 70.26 | 59.14 | 26.61 | 41.71 | 53.14 | 54.64 | 40.37 | 30.95 | 36.74 |
| **20%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 18.86 | 52.24 | 42.52 | 22.80 | 30.53 | 35.66 | 35.74 | 24.85 | 19.73 | 35.34 |
| **ACSS-2** | 18.93 | 53.36 | 42.10 | 23.31 | 30.61 | 36.94 | 35.53 | 27.02 | 19.64 | 35.60 |
| **ACSS-3** | 18.89 | 52.78 | 41.73 | 23.52 | 30.51 | 36.72 | 35.49 | 25.62 | 19.74 | 36.07 |
| **30%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 17.68 | 35.54 | 30.21 | 20.49 | 22.39 | 27.99 | 28.87 | 22.61 | 21.29 | 32.01 |
| **ACSS-2** | 17.75 | 35.74 | 30.81 | 20.93 | 22.57 | 27.51 | 28.93 | 23.18 | 21.10 | 32.14 |
| **ACSS-3** | 17.28 | 35.53 | 30.73 | 20.78 | 22.65 | 27.85 | 28.87 | 22.87 | 21.55 | 32.00 |
| **40%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 18.66 | 41.05 | 27.93 | 17.59 | 25.73 | 24.47 | 26.02 | 18.51 | 21.55 | 25.29 |
| **ACSS-2** | 18.70 | 41.25 | 28.40 | 17.35 | 25.69 | 24.71 | 26.44 | 18.60 | 22.06 | 26.27 |
| **ACSS-3** | 18.73 | 40.46 | 28.04 | 17.29 | 25.65 | 24.38 | 26.67 | 18.64 | 22.11 | 26.37 |
| **50%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 18.48 | 33.03 | 30.67 | 23.67 | 22.98 | 21.93 | 23.92 | 17.95 | 16.86 | 26.07 |
| **ACSS-2** | 18.57 | 33.25 | 30.61 | 23.70 | 23.07 | 22.10 | 23.92 | 18.04 | 16.96 | 26.26 |
| **ACSS-3** | 18.59 | 33.18 | 30.39 | 23.62 | 23.14 | 22.32 | 23.99 | 17.89 | 16.80 | 26.24 |

## Table 175. Timing Results - ACSE (15 Nodes)

| Timing Results- ACSE (15 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 858.37 | 700.01 | 438.02 | 391.04 | 498.49 | 459.84 | 537.02 | 596.54 | 876.43 | 692.22 |
| **ACSE-2** | 862.89 | 700.82 | 434.66 | 390.82 | 503.51 | 465.04 | 524.93 | 595.00 | 911.95 | 685.30 |
| **ACSE-3** | 868.01 | 707.43 | 437.61 | 392.02 | 499.16 | 461.58 | 524.32 | 601.45 | 875.91 | 700.40 |
| **10%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 655.37 | 521.53 | 297.60 | 332.60 | 350.47 | 335.11 | 401.81 | 411.07 | 656.85 | 562.37 |
| **ACSE-2** | 679.62 | 522.85 | 308.96 | 345.36 | 363.84 | 348.10 | 425.22 | 426.40 | 657.42 | 585.09 |
| **ACSE-3** | 679.03 | 523.14 | 308.86 | 346.81 | 364.54 | 349.04 | 419.04 | 426.71 | 658.80 | 583.28 |
| **20%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 595.63 | 493.81 | 290.57 | 267.96 | 289.83 | 299.56 | 267.44 | 385.92 | 552.47 | 477.83 |
| **ACSE-2** | 577.56 | 491.13 | 282.96 | 261.37 | 284.00 | 292.95 | 256.99 | 371.99 | 533.28 | 484.01 |
| **ACSE-3** | 571.58 | 475.37 | 279.07 | 257.78 | 279.01 | 288.10 | 257.43 | 370.69 | 533.33 | 468.64 |
| **30%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 419.18 | 318.65 | 232.81 | 203.16 | 278.16 | 243.47 | 246.02 | 274.01 | 410.14 | 328.93 |
| **ACSE-2** | 417.38 | 316.52 | 232.46 | 202.86 | 278.61 | 243.13 | 245.83 | 273.75 | 408.61 | 328.79 |
| **ACSE-3** | 418.54 | 316.73 | 232.67 | 203.18 | 279.61 | 243.59 | 246.68 | 273.90 | 409.46 | 328.49 |
| **40%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 344.91 | 340.35 | 184.00 | 174.53 | 240.75 | 221.08 | 242.89 | 289.99 | 327.97 | 294.37 |
| **ACSE-2** | 375.70 | 370.96 | 200.65 | 190.42 | 262.78 | 241.49 | 264.96 | 315.10 | 356.43 | 320.22 |
| **ACSE-3** | 343.94 | 352.53 | 184.31 | 177.28 | 248.59 | 227.09 | 243.08 | 288.83 | 334.67 | 301.44 |
| **50%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSE-1** | 288.98 | 282.40 | 174.14 | 178.23 | 206.39 | 200.40 | 204.59 | 220.81 | 313.50 | 280.53 |
| **ACSE-2** | 289.19 | 282.13 | 174.06 | 178.13 | 205.75 | 200.74 | 205.87 | 221.03 | 314.41 | 280.29 |
| **ACSE-3** | 289.34 | 281.94 | 173.75 | 178.27 | 206.69 | 200.25 | 205.14 | 220.20 | 312.79 | 280.99 |

**Table 176. Timing Results - ACSS (15 Nodes)**

| Timing Results - ACSS (15 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 494.62 | 385.79 | 252.89 | 213.19 | 261.39 | 266.49 | 292.76 | 300.85 | 447.88 | 398.34 |
| **ACSS-2** | 513.65 | 391.76 | 252.18 | 211.74 | 286.68 | 240.85 | 272.23 | 319.26 | 467.39 | 398.68 |
| **ACSS-3** | 471.43 | 407.67 | 233.74 | 217.74 | 297.53 | 236.16 | 270.42 | 312.07 | 482.14 | 343.36 |
| **10%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 428.21 | 338.31 | 199.45 | 216.47 | 233.60 | 218.04 | 284.27 | 281.15 | 429.52 | 370.86 |
| **ACSS-2** | 441.43 | 334.92 | 198.12 | 215.24 | 237.38 | 227.51 | 265.78 | 278.59 | 443.33 | 349.60 |
| **ACSS-3** | 419.07 | 336.32 | 206.22 | 222.49 | 230.66 | 225.83 | 257.55 | 270.56 | 447.63 | 355.69 |
| **20%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 380.30 | 306.72 | 187.31 | 176.13 | 184.21 | 195.36 | 173.22 | 255.83 | 364.16 | 304.92 |
| **ACSS-2** | 397.64 | 312.49 | 192.77 | 172.89 | 187.50 | 201.61 | 176.81 | 253.74 | 386.87 | 309.66 |
| **ACSS-3** | 383.89 | 307.92 | 188.36 | 169.89 | 182.27 | 193.90 | 171.26 | 258.20 | 373.48 | 314.50 |
| **30%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 298.22 | 227.04 | 161.18 | 138.56 | 188.71 | 172.29 | 167.82 | 188.94 | 282.43 | 231.45 |
| **ACSS-2** | 298.65 | 219.61 | 162.12 | 137.51 | 195.82 | 168.69 | 167.68 | 193.00 | 288.97 | 225.83 |
| **ACSS-3** | 300.91 | 224.47 | 162.27 | 140.48 | 190.78 | 168.38 | 163.98 | 193.11 | 291.72 | 231.55 |
| **40%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 252.39 | 247.14 | 127.91 | 119.90 | 165.73 | 153.22 | 170.06 | 198.15 | 231.28 | 203.78 |
| **ACSS-2** | 252.42 | 244.99 | 130.04 | 118.09 | 168.29 | 155.99 | 176.37 | 206.35 | 233.55 | 207.13 |
| **ACSS-3** | 253.05 | 245.55 | 129.90 | 120.51 | 169.51 | 156.58 | 172.95 | 203.67 | 232.43 | 203.29 |
| **50%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **ACSS-1** | 207.44 | 200.45 | 123.17 | 127.76 | 146.52 | 144.22 | 147.56 | 158.77 | 231.72 | 197.27 |
| **ACSS-2** | 204.27 | 202.23 | 121.40 | 125.07 | 146.55 | 140.80 | 144.66 | 158.43 | 229.02 | 194.90 |
| **ACSS-3** | 206.10 | 201.74 | 121.76 | 126.60 | 147.00 | 142.45 | 143.88 | 157.24 | 227.77 | 197.07 |

**Table 177. Timing Results - DACSE (10 Nodes)**

| Timing Results - DACSE (10 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 14.07 | 19.91 | 18.67 | 13.19 | 18.20 | 16.06 | 16.12 | 17.61 | 15.29 | 17.60 |
| **DACSE-2** | 13.92 | 19.85 | 18.71 | 13.05 | 17.93 | 15.94 | 16.80 | 17.77 | 15.05 | 17.67 |
| **DACSE-3** | 14.01 | 20.13 | 18.72 | 13.00 | 18.03 | 16.02 | 16.21 | 17.64 | 15.26 | 17.93 |
| **10%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 14.06 | 17.50 | 17.25 | 12.30 | 15.21 | 17.22 | 15.59 | 17.93 | 15.99 | 13.36 |
| **DACSE-2** | 13.99 | 17.45 | 17.23 | 12.22 | 15.11 | 17.11 | 15.69 | 17.86 | 15.95 | 13.35 |
| **DACSE-3** | 14.03 | 17.63 | 17.36 | 12.26 | 15.17 | 17.17 | 15.57 | 17.91 | 15.97 | 13.43 |
| **20%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 10.82 | 15.00 | 14.55 | 10.10 | 12.19 | 12.39 | 10.81 | 13.01 | 9.20 | 13.66 |
| **DACSE-2** | 10.82 | 14.97 | 14.51 | 10.09 | 12.20 | 12.36 | 12.41 | 12.98 | 9.16 | 13.66 |
| **DACSE-3** | 10.85 | 15.27 | 14.65 | 10.31 | 12.40 | 12.64 | 10.84 | 13.05 | 9.22 | 13.73 |
| **30%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 8.88 | 10.78 | 10.22 | 8.66 | 8.90 | 10.54 | 9.69 | 10.87 | 10.65 | 11.16 |
| **DACSE-2** | 8.85 | 10.76 | 10.19 | 8.62 | 8.88 | 10.50 | 9.73 | 10.84 | 10.62 | 11.15 |
| **DACSE-3** | 9.17 | 11.17 | 10.62 | 8.96 | 9.15 | 10.93 | 10.00 | 11.24 | 10.99 | 11.58 |
| **40%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 8.39 | 10.30 | 8.74 | 9.15 | 9.35 | 8.60 | 9.28 | 10.73 | 10.58 | 10.57 |
| **DACSE-2** | 8.62 | 11.52 | 9.73 | 8.26 | 10.50 | 8.73 | 9.59 | 9.64 | 11.18 | 10.81 |
| **DACSE-3** | 8.65 | 11.57 | 9.77 | 8.29 | 10.51 | 8.75 | 9.51 | 10.86 | 10.89 | 10.80 |
| **50%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 8.32 | 9.82 | 9.98 | 9.06 | 9.12 | 9.98 | 8.32 | 9.42 | 8.13 | 9.57 |
| **DACSE-2** | 8.07 | 9.48 | 9.63 | 8.82 | 8.86 | 9.64 | 9.04 | 9.17 | 7.90 | 9.29 |
| **DACSE-3** | 8.33 | 9.79 | 9.96 | 9.07 | 9.12 | 10.01 | 8.33 | 9.44 | 8.16 | 9.57 |

**Table 178. Timing Results - DACSS (10 Nodes)**

| Timing Results - DACSS (10 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 7.19 | 8.59 | 7.68 | 5.87 | 10.40 | 5.98 | 7.08 | 8.64 | 6.76 | 7.91 |
| **DACSS-2** | 7.59 | 8.60 | 7.74 | 6.05 | 10.50 | 6.09 | 7.43 | 8.83 | 7.10 | 8.19 |
| **DACSS-3** | 7.12 | 8.64 | 7.76 | 6.09 | 10.52 | 5.92 | 7.26 | 8.79 | 6.59 | 7.91 |
| **10%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 6.98 | 7.31 | 6.86 | 5.23 | 7.80 | 6.91 | 6.40 | 8.31 | 7.78 | 6.25 |
| **DACSS-2** | 6.92 | 7.50 | 6.81 | 5.31 | 7.90 | 7.15 | 6.53 | 8.31 | 7.90 | 6.11 |
| **DACSS-3** | 7.02 | 7.49 | 7.00 | 5.28 | 7.72 | 7.15 | 6.46 | 8.31 | 7.66 | 6.21 |
| **20%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 5.90 | 6.57 | 6.82 | 4.64 | 6.77 | 5.14 | 4.79 | 5.71 | 4.26 | 6.27 |
| **DACSS-2** | 5.96 | 6.95 | 7.01 | 4.59 | 6.73 | 5.33 | 4.79 | 6.19 | 4.19 | 6.32 |
| **DACSS-3** | 5.89 | 6.83 | 6.94 | 4.65 | 6.78 | 5.37 | 4.69 | 5.91 | 4.17 | 6.46 |
| **30%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 4.51 | 4.76 | 4.08 | 3.82 | 4.62 | 4.47 | 4.50 | 4.96 | 5.12 | 5.24 |
| **DACSS-2** | 4.53 | 4.83 | 4.39 | 3.90 | 4.64 | 4.51 | 4.54 | 5.31 | 5.18 | 5.17 |
| **DACSS-3** | 4.47 | 4.87 | 4.24 | 3.98 | 4.70 | 4.47 | 4.57 | 5.12 | 5.26 | 5.26 |
| **40%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 3.91 | 4.59 | 3.90 | 3.49 | 5.07 | 2.98 | 3.38 | 3.89 | 5.11 | 4.36 |
| **DACSS-2** | 3.83 | 4.65 | 3.98 | 3.53 | 5.11 | 3.11 | 3.44 | 3.87 | 5.19 | 4.67 |
| **DACSS-3** | 3.92 | 4.65 | 3.85 | 3.50 | 5.14 | 3.01 | 3.46 | 3.95 | 5.23 | 4.70 |
| **50%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 4.16 | 4.43 | 4.53 | 4.53 | 4.83 | 4.59 | 4.01 | 4.86 | 3.84 | 4.89 |
| **DACSS-2** | 4.20 | 4.49 | 4.52 | 4.61 | 4.81 | 4.62 | 4.00 | 4.90 | 3.84 | 4.84 |
| **DACSS-3** | 3.77 | 3.93 | 3.98 | 4.08 | 4.22 | 3.98 | 3.49 | 4.28 | 3.40 | 4.26 |

## Table 179. Timing Results - DACSE (15 Nodes)

| Timing Results - DACSE (15 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 109.57 | 100.26 | 77.08 | 77.19 | 82.57 | 94.64 | 104.22 | 79.68 | 106.53 | 111.43 |
| **DACSE-2** | 109.18 | 99.86 | 76.00 | 78.44 | 82.31 | 94.07 | 103.45 | 79.68 | 106.14 | 110.52 |
| **DACSE-3** | 109.07 | 101.26 | 77.12 | 78.79 | 83.07 | 94.17 | 103.57 | 79.35 | 106.58 | 111.81 |
| **10%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 92.11 | 80.46 | 58.09 | 69.73 | 65.74 | 77.46 | 84.04 | 67.53 | 94.01 | 99.71 |
| **DACSE-2** | 91.39 | 79.57 | 57.75 | 69.29 | 65.53 | 76.95 | 83.81 | 68.08 | 95.89 | 103.65 |
| **DACSE-3** | 91.96 | 80.18 | 58.04 | 69.75 | 65.88 | 77.30 | 84.17 | 67.68 | 94.11 | 99.26 |
| **20%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 82.43 | 83.91 | 54.35 | 56.09 | 56.59 | 67.18 | 62.53 | 60.15 | 80.99 | 77.65 |
| **DACSE-2** | 78.72 | 78.41 | 51.82 | 53.36 | 54.01 | 64.42 | 60.08 | 57.25 | 77.40 | 74.13 |
| **DACSE-3** | 78.59 | 74.71 | 51.98 | 53.83 | 54.30 | 64.68 | 60.35 | 57.59 | 77.65 | 74.44 |
| **30%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 63.98 | 57.84 | 49.12 | 51.60 | 54.80 | 59.38 | 71.71 | 47.62 | 65.36 | 63.17 |
| **DACSE-2** | 61.07 | 55.25 | 47.09 | 49.47 | 52.06 | 56.56 | 67.11 | 45.49 | 62.29 | 60.14 |
| **DACSE-3** | 61.31 | 55.50 | 47.29 | 49.71 | 52.31 | 56.89 | 67.35 | 45.59 | 62.77 | 60.30 |
| **40%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 52.12 | 54.73 | 37.43 | 40.57 | 46.02 | 52.76 | 53.62 | 47.82 | 53.44 | 56.43 |
| **DACSE-2** | 57.08 | 59.94 | 41.09 | 44.51 | 50.66 | 57.66 | 58.22 | 52.39 | 58.32 | 61.82 |
| **DACSE-3** | 51.97 | 55.35 | 37.56 | 40.62 | 46.04 | 52.76 | 53.38 | 47.71 | 53.39 | 57.82 |
| **50%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSE-1** | 51.98 | 51.47 | 38.67 | 41.44 | 40.72 | 47.33 | 51.97 | 39.55 | 50.57 | 59.10 |
| **DACSE-2** | 49.22 | 48.89 | 37.33 | 39.88 | 39.06 | 45.70 | 50.54 | 39.17 | 47.99 | 56.17 |
| **DACSE-3** | 52.03 | 51.97 | 39.40 | 42.09 | 41.23 | 47.89 | 52.78 | 40.12 | 50.83 | 59.27 |

**Table 180. Timing Results - DACSS (15 Nodes)**

| Timing Results - DACSS (15 Nodes) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **0%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 33.63 | 36.60 | 25.39 | 23.39 | 23.41 | 33.83 | 44.05 | 24.77 | 38.68 | 34.17 |
| **DACSS-2** | 36.07 | 32.18 | 25.21 | 24.91 | 22.90 | 36.35 | 44.45 | 23.88 | 35.13 | 35.49 |
| **DACSS-3** | 35.44 | 33.98 | 24.66 | 25.12 | 25.00 | 37.32 | 45.83 | 24.58 | 37.51 | 35.94 |
| **10%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 33.82 | 29.58 | 21.41 | 24.80 | 21.54 | 33.63 | 36.84 | 25.20 | 37.73 | 36.22 |
| **DACSS-2** | 29.54 | 26.40 | 19.18 | 24.18 | 19.42 | 28.85 | 34.43 | 21.39 | 34.49 | 32.86 |
| **DACSS-3** | 34.96 | 29.58 | 22.36 | 27.03 | 21.78 | 35.77 | 38.99 | 23.98 | 39.39 | 36.81 |
| **20%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 30.75 | 31.85 | 19.19 | 21.24 | 20.05 | 28.34 | 27.82 | 22.19 | 32.11 | 319.24 |
| **DACSS-2** | 31.73 | 32.60 | 20.28 | 21.49 | 21.26 | 30.26 | 29.29 | 21.91 | 32.80 | 312.78 |
| **DACSS-3** | 31.53 | 30.42 | 17.94 | 21.96 | 20.27 | 29.26 | 28.20 | 23.32 | 33.33 | 344.11 |
| **30%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 24.82 | 23.42 | 20.10 | 21.65 | 19.28 | 24.74 | 87.32 | 17.07 | 23.92 | 23.12 |
| **DACSS-2** | 25.45 | 23.23 | 19.62 | 21.38 | 20.12 | 24.37 | 33.25 | 17.42 | 24.56 | 23.45 |
| **DACSS-3** | 25.95 | 23.10 | 19.87 | 22.16 | 19.14 | 24.65 | 34.54 | 17.61 | 25.01 | 23.45 |
| **40%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 23.86 | 23.91 | 16.19 | 16.19 | 17.90 | 23.31 | 25.63 | 19.57 | 21.65 | 24.13 |
| **DACSS-2** | 23.51 | 23.62 | 16.14 | 16.43 | 17.37 | 23.79 | 25.75 | 19.49 | 22.76 | 24.17 |
| **DACSS-3** | 23.62 | 23.92 | 16.05 | 16.68 | 17.75 | 22.72 | 25.96 | 19.43 | 21.96 | 24.02 |
| **50%** | **File 1** | **File 2** | **File 3** | **File 4** | **File 5** | **File 6** | **File 7** | **File 8** | **File 9** | **File 10** |
| **DACSS-1** | 20.20 | 21.94 | 15.50 | 16.64 | 15.00 | 20.99 | 26.18 | 15.35 | 20.47 | 25.88 |
| **DACSS-2** | 19.38 | 21.91 | 15.52 | 17.18 | 15.25 | 20.69 | 26.30 | 15.35 | 20.04 | 23.59 |
| **DACSS-3** | 20.03 | 21.99 | 15.35 | 16.27 | 15.24 | 20.81 | 25.97 | 15.78 | 20.83 | 24.76 |

## Appendix L: Convergence Charts (Average of all ten test files)

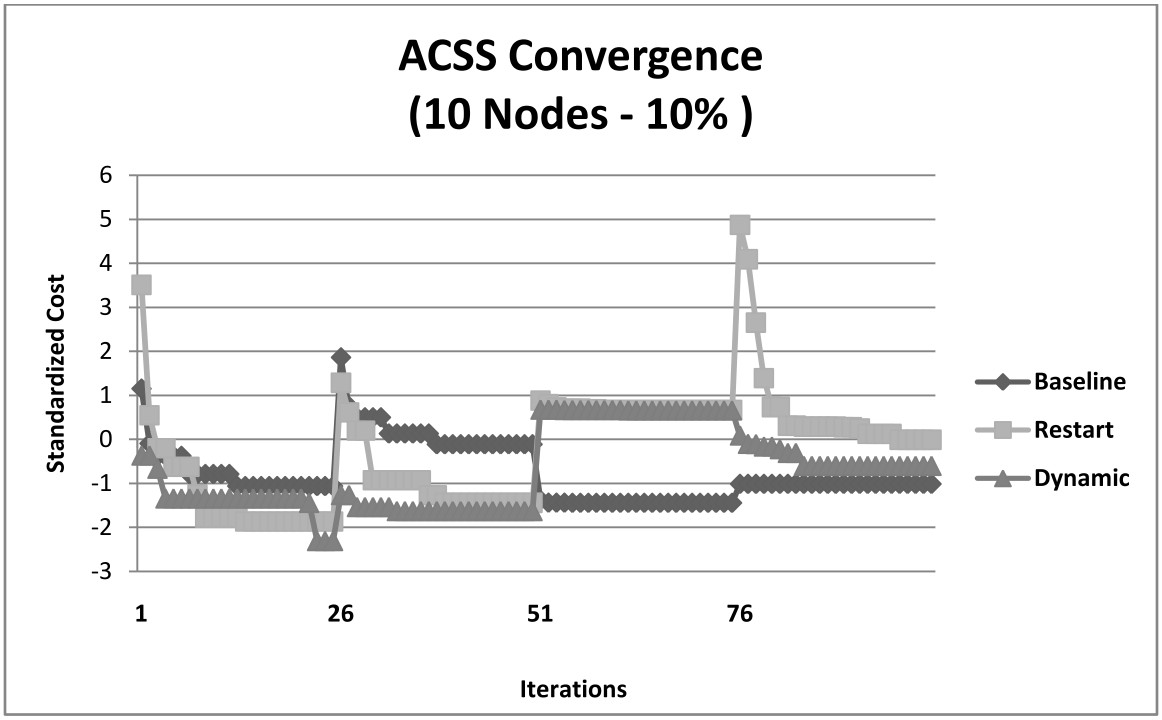


**Figure 46. ACSS Average Convergence (10 Nodes - 10% Change)**

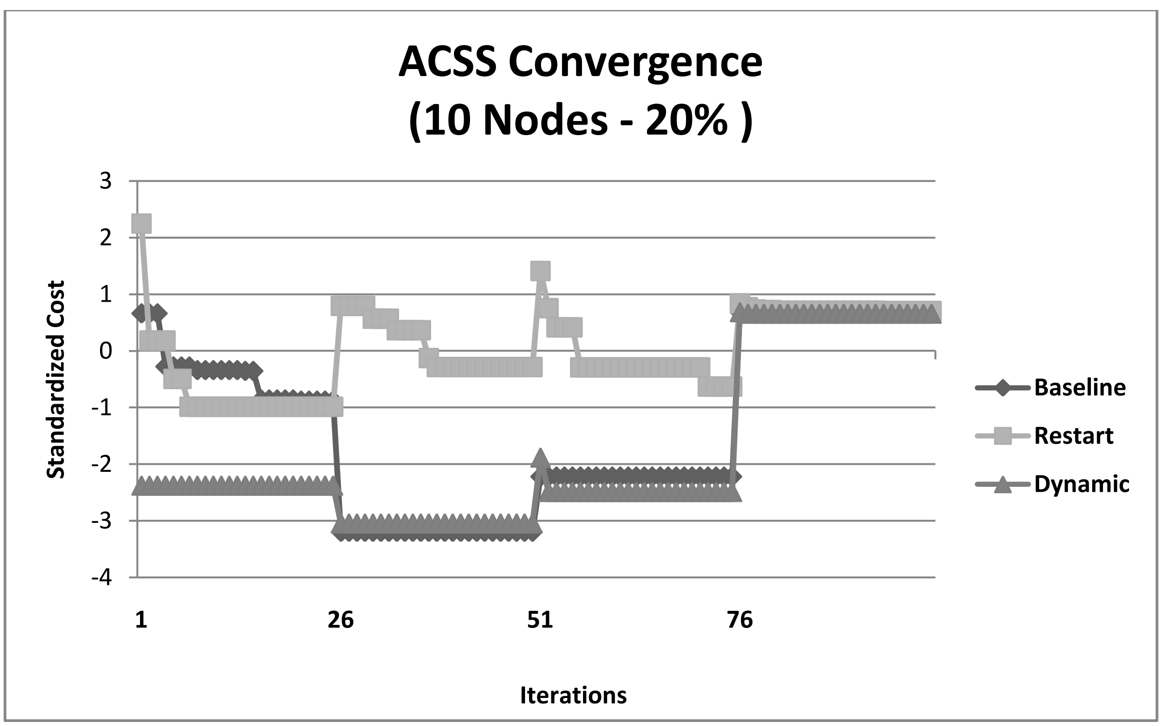


**Figure 47. ACSS Average Convergence (10 Nodes - 20% Change)**

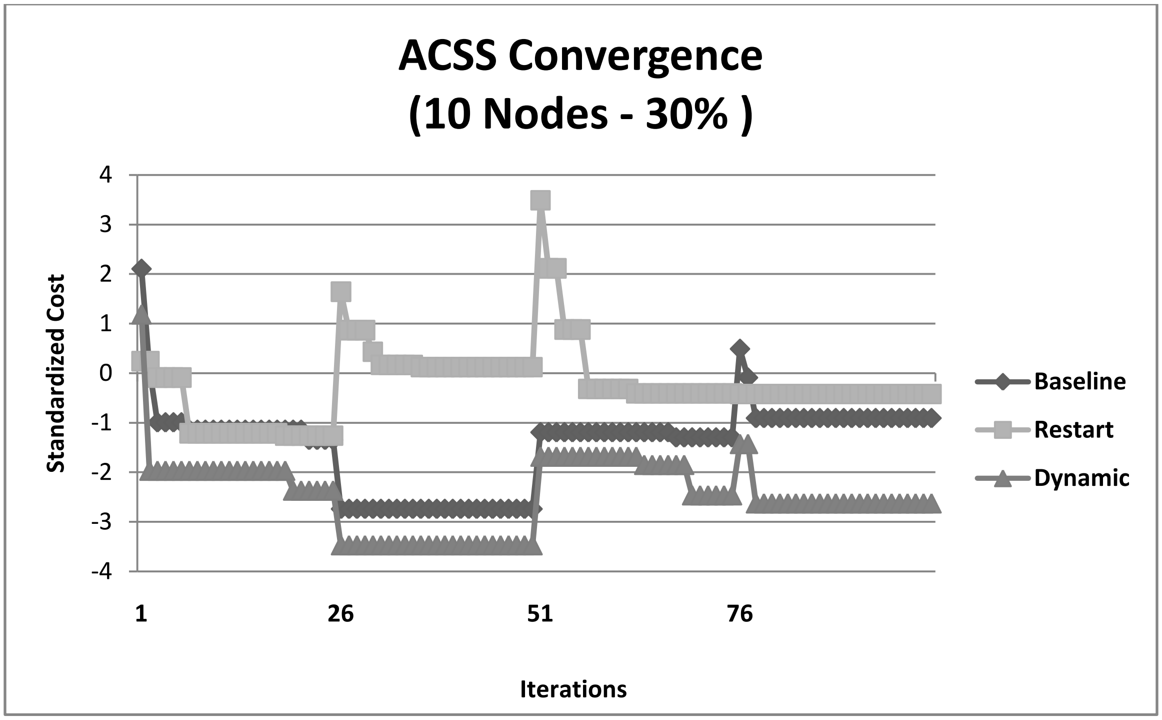


**Figure 48. ACSS Average Convergence (10 Nodes - 30% Change)**

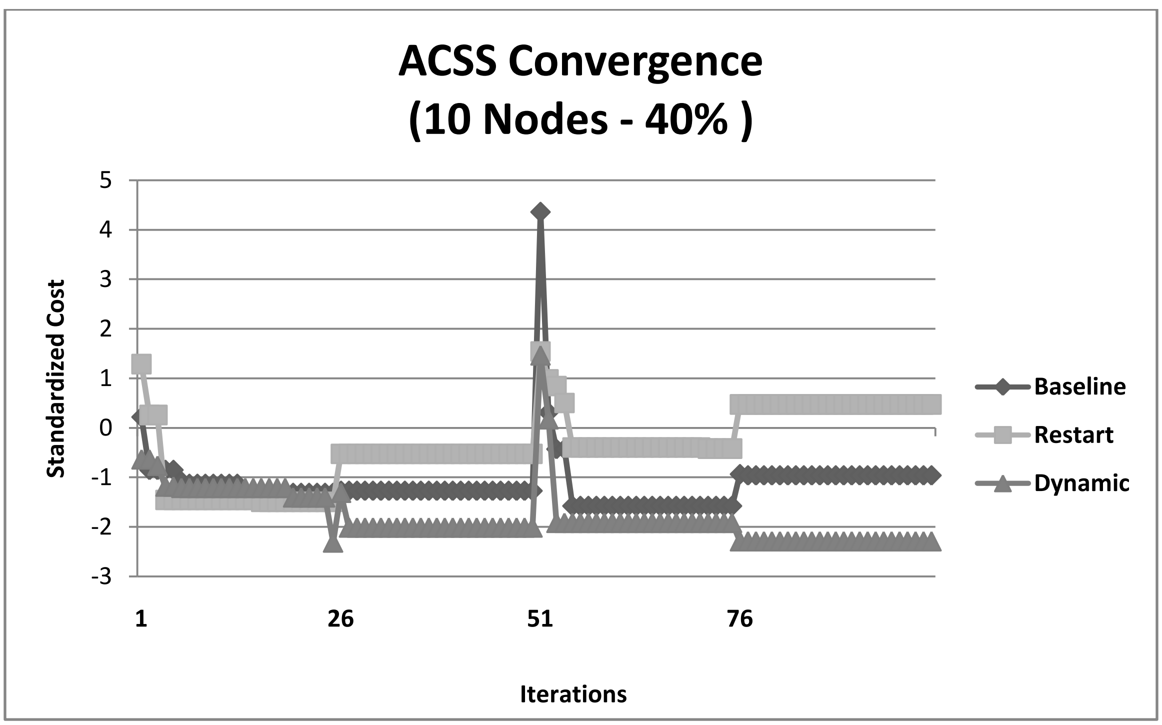


**Figure 49. ACSS Average Convergence (10 Nodes - 40% Change)**

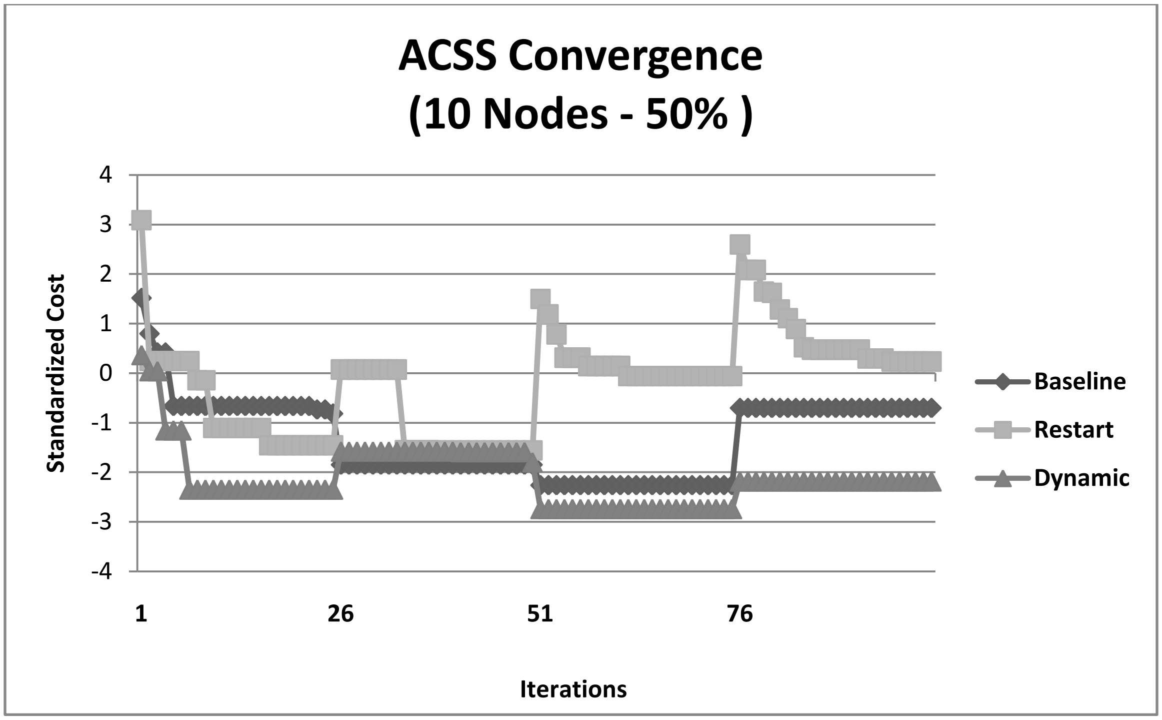


Figure 50. ACSS Average Convergence (10 Nodes - 50% Change)

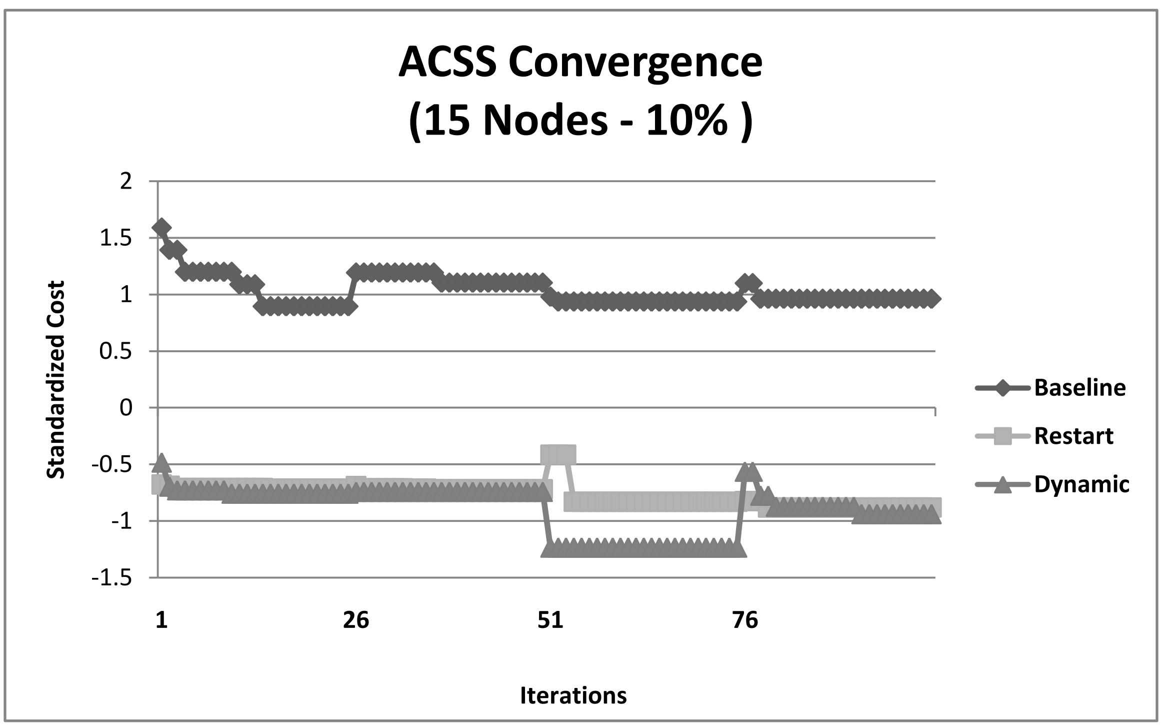


Figure 51. ACSS Average Convergence (15 Nodes - 10% Change)

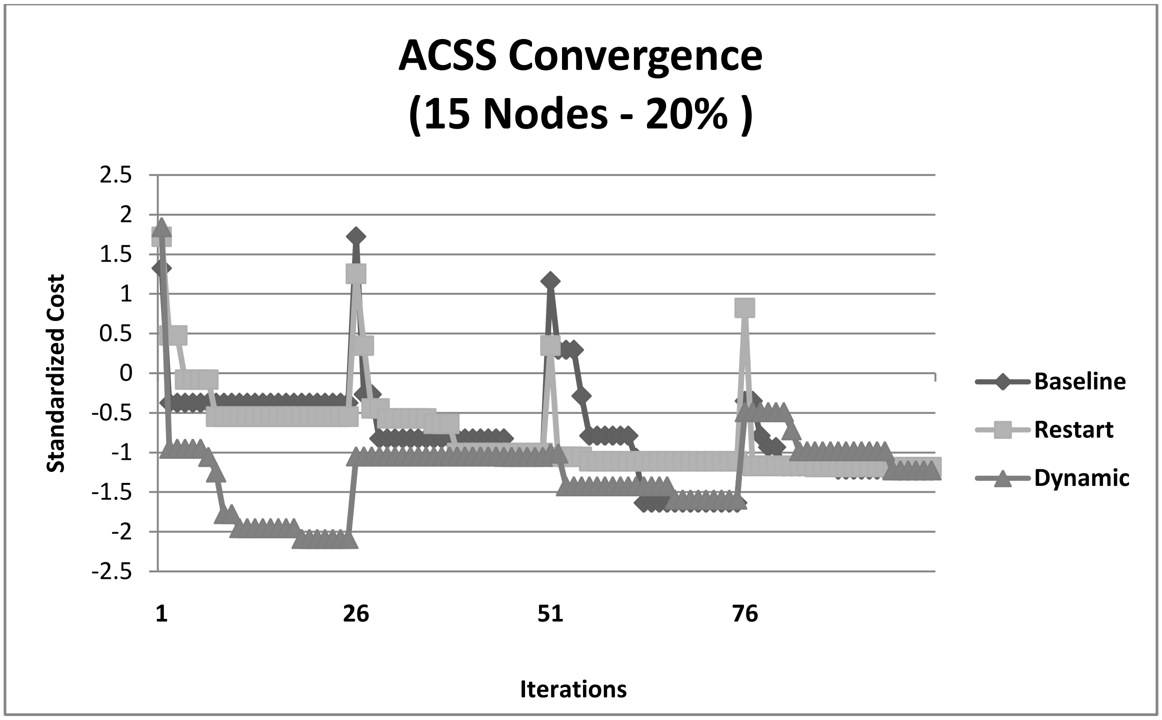


**Figure 52. ACSS Average Convergence (15 Nodes - 20% Change)**

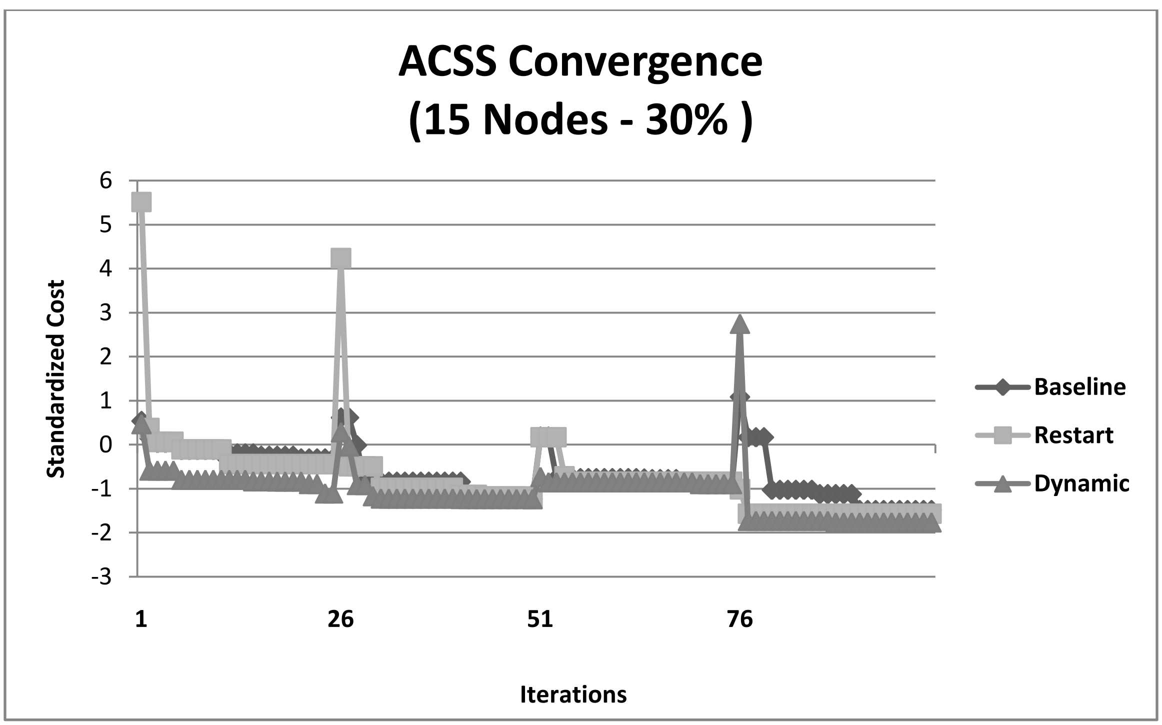


**Figure 53. ACSS Average Convergence (15 Nodes - 30% Change)**

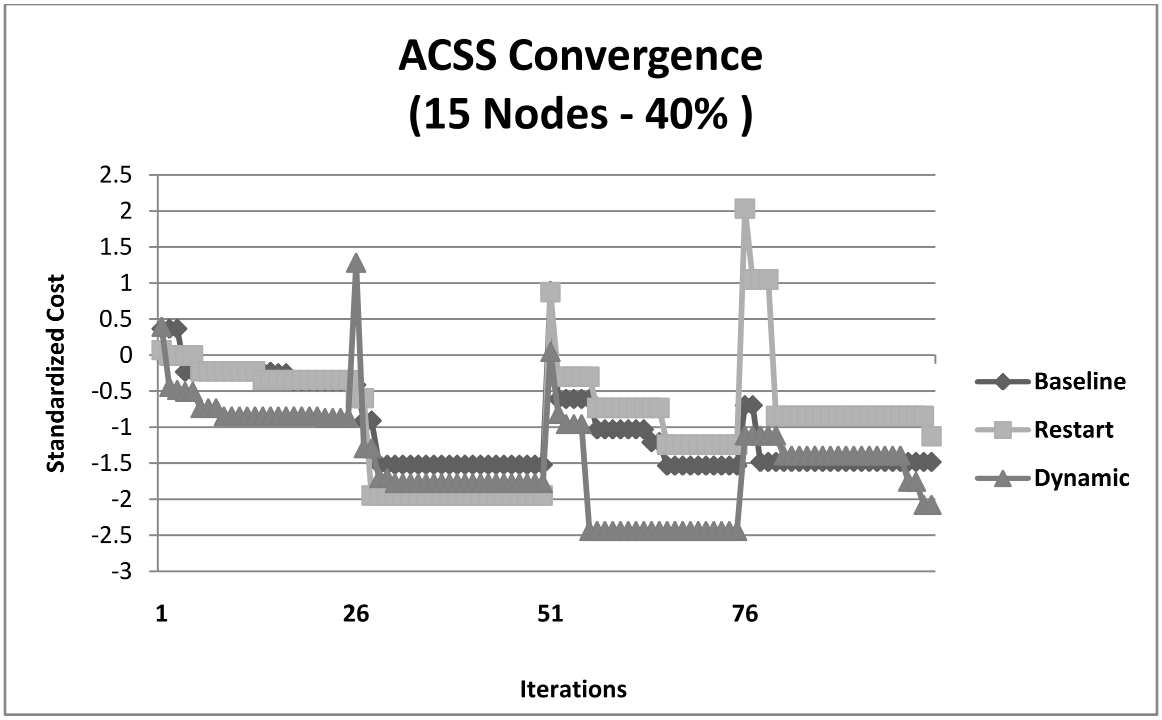


**Figure 54. ACSS Average Convergence (15 Nodes - 40% Change)**

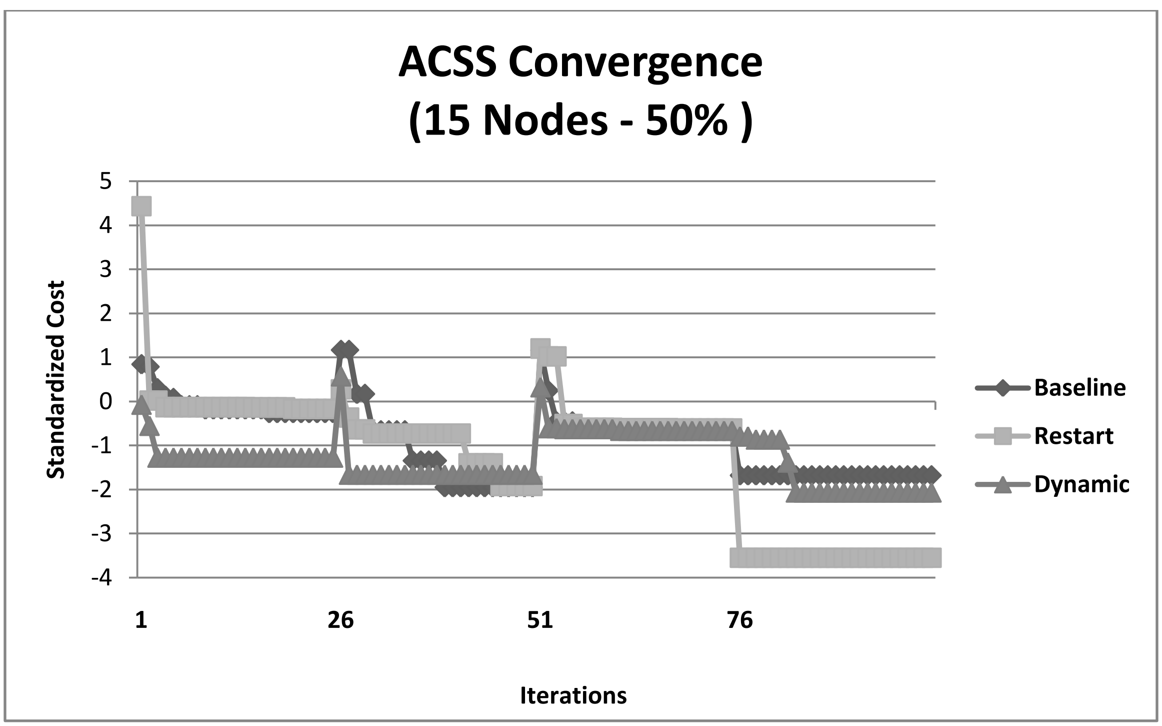


**Figure 55. ACSS Average Convergence (15 Nodes - 50% Change)**

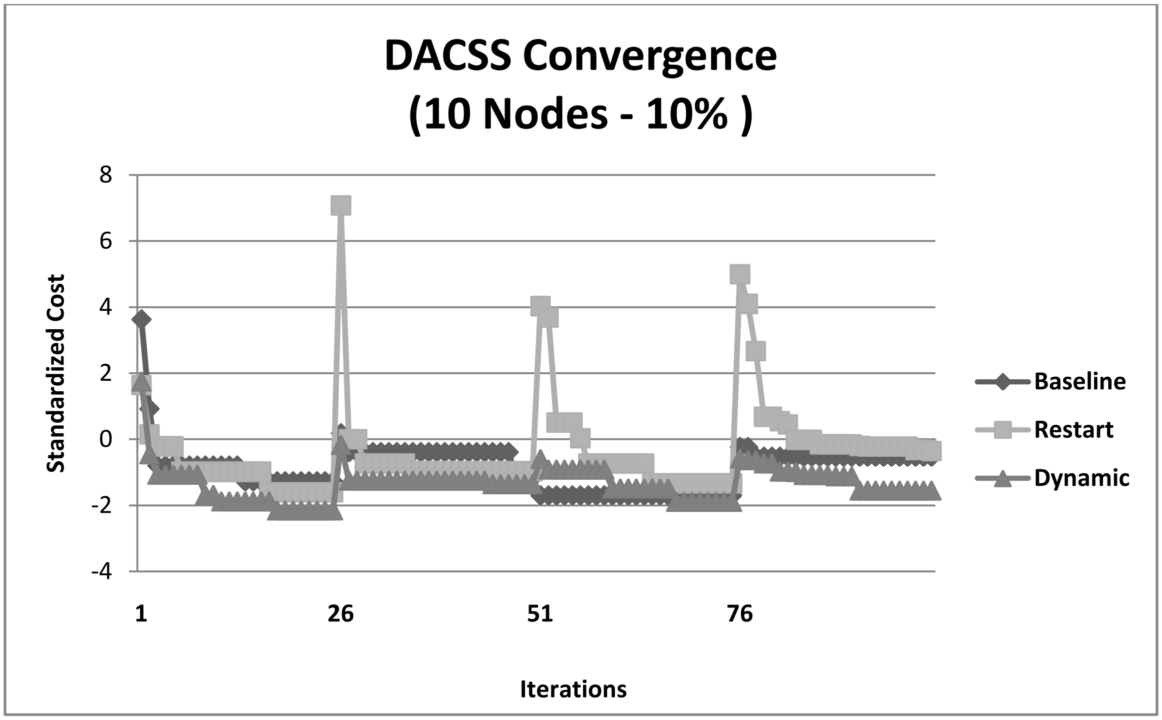


**Figure 56. DACSS Average Convergence (10 Nodes - 10% Change)**

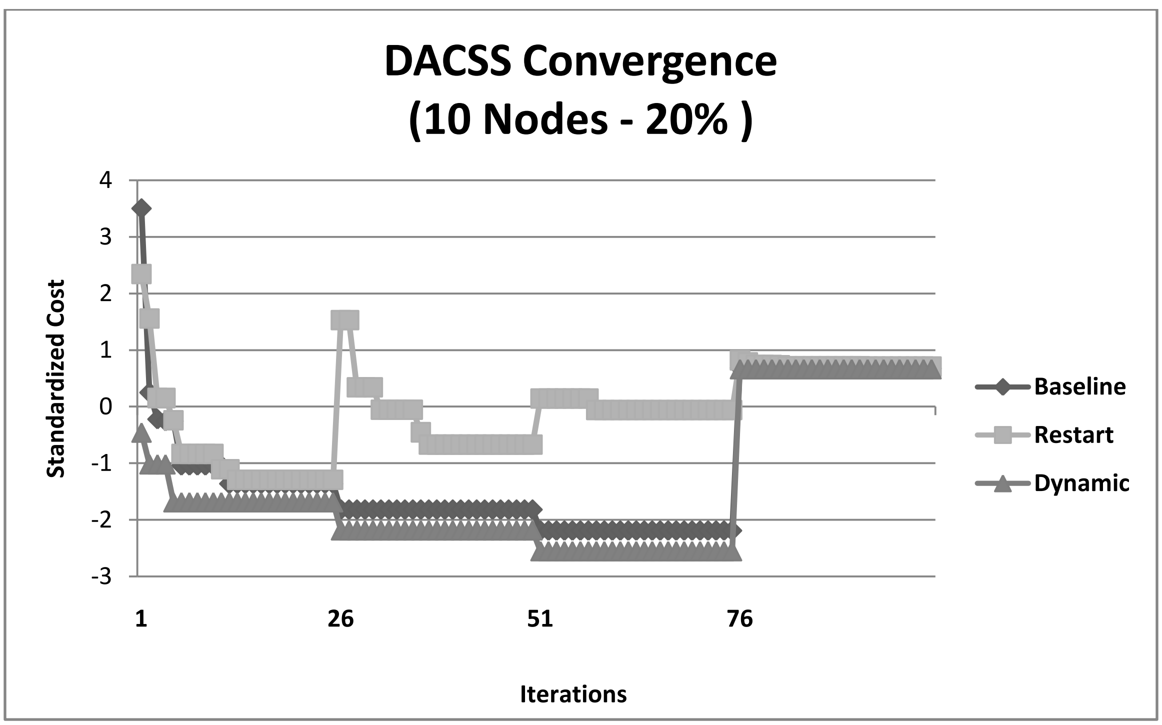


**Figure 57. DACSS Average Convergence (10 Nodes - 20% Change)**

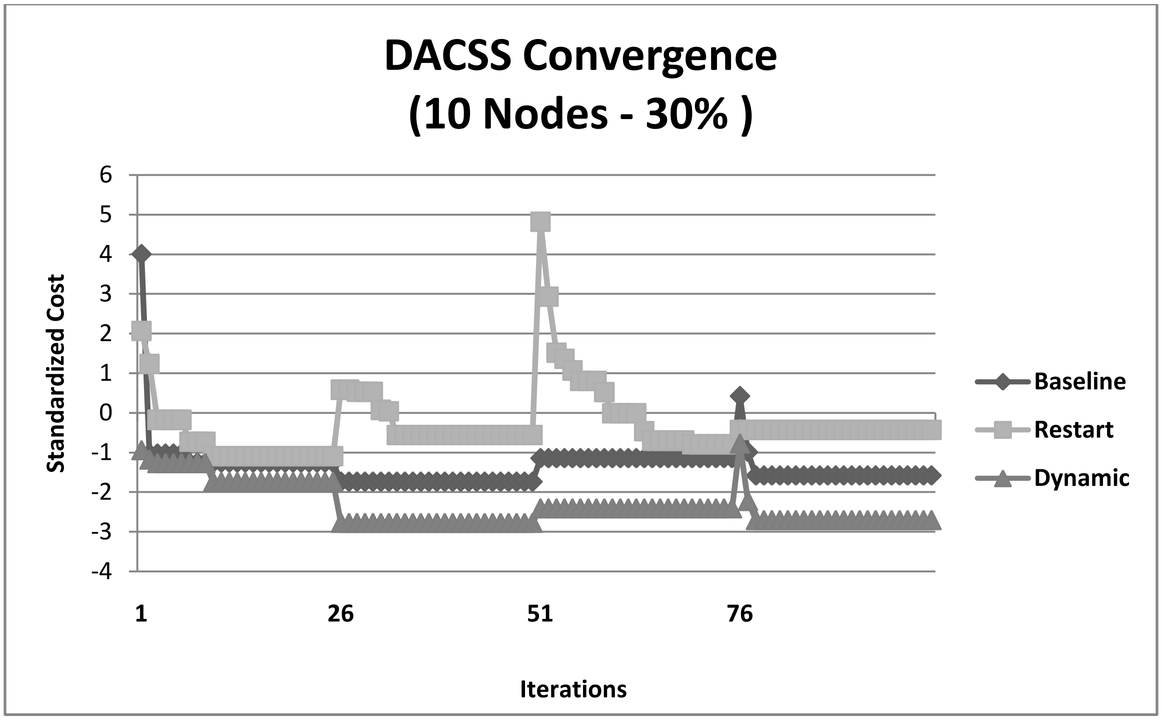

**Figure 58. DACSS Average Convergence (10 Nodes - 30% Change)**

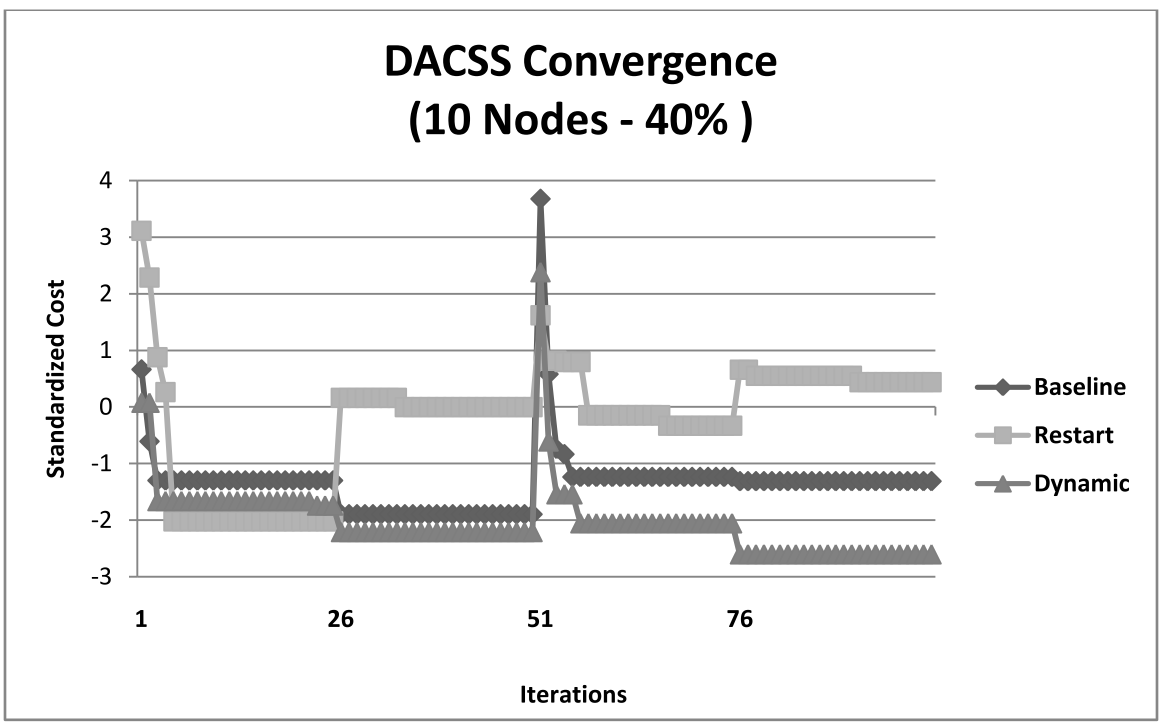

**Figure 59. DACSS Average Convergence (10 Nodes - 40% Change)**

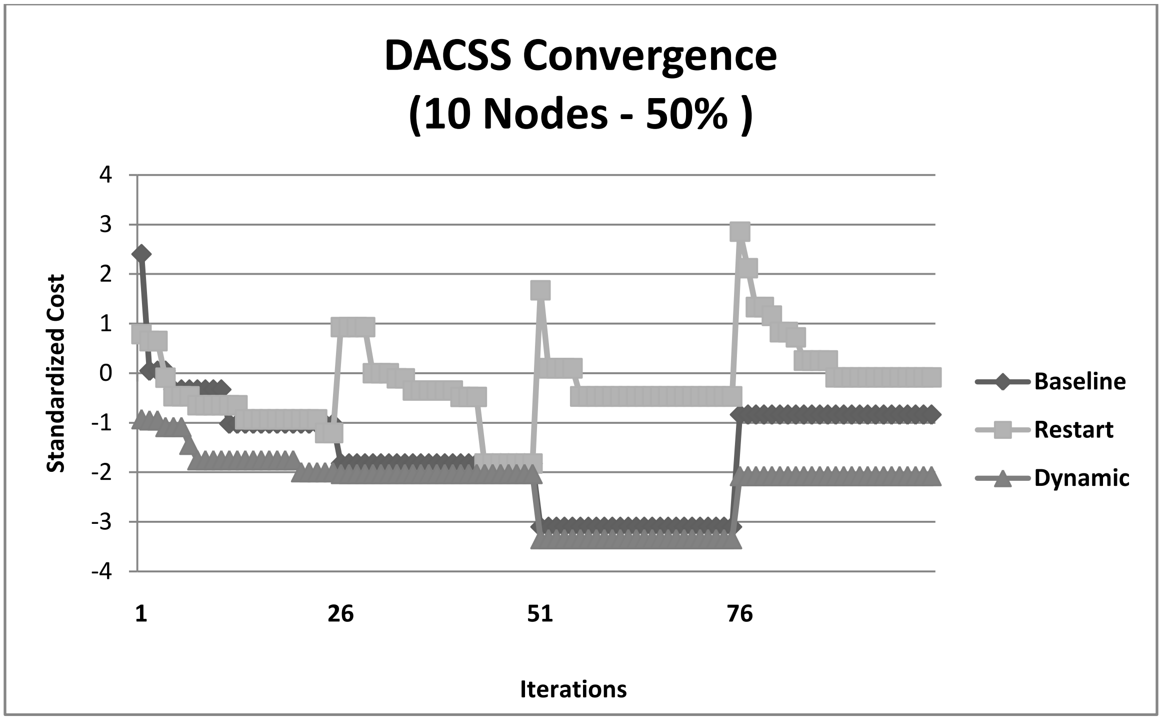


**Figure 60. DACSS Average Convergence (10 Nodes - 50% Change)**

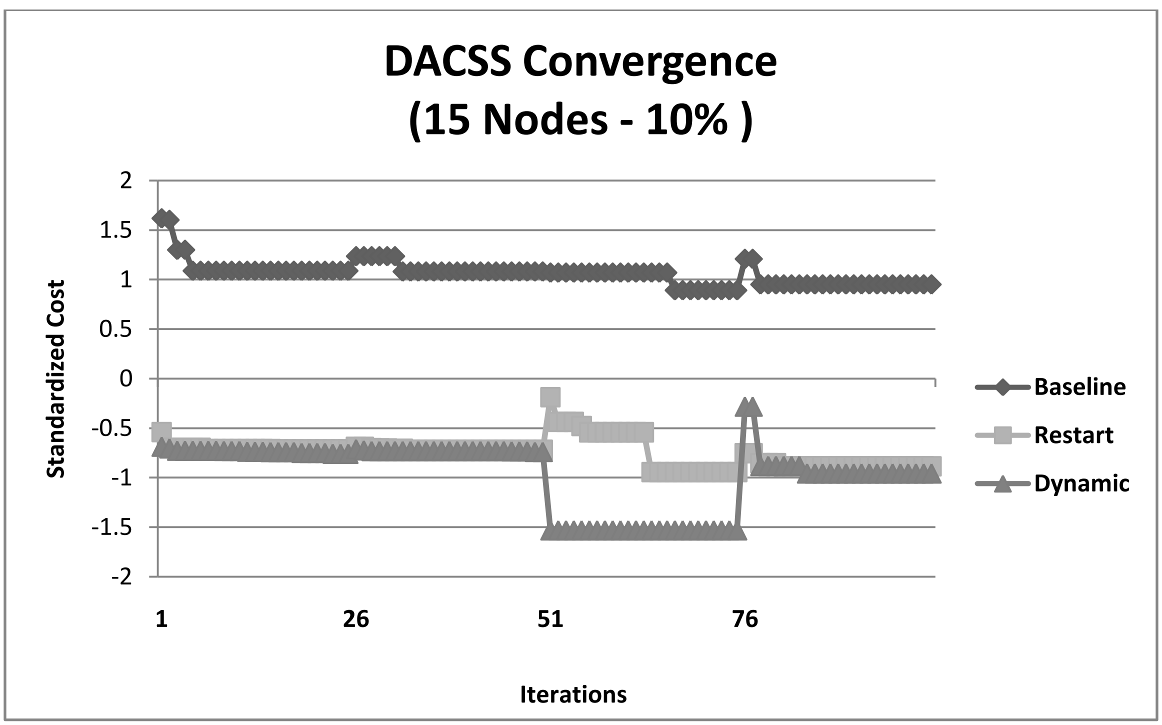


**Figure 61. DACSS Average Convergence (15 Nodes - 10% Change)**

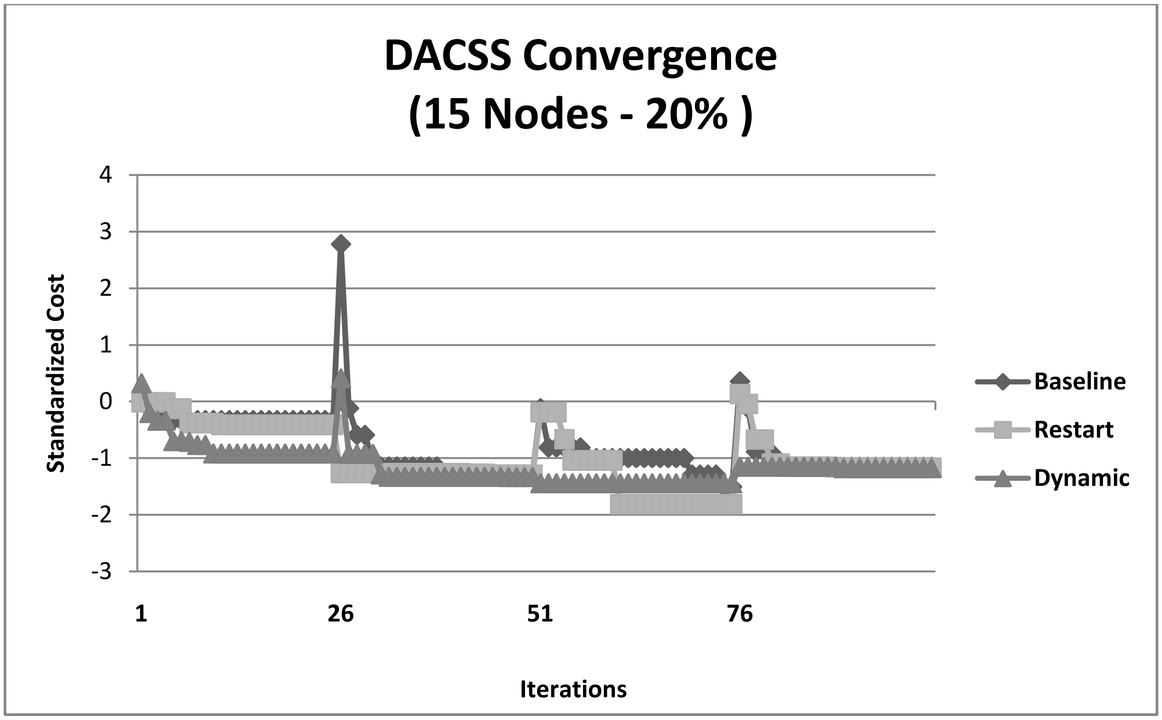


**Figure 62. DACSS Average Convergence (15 Nodes - 20% Change)**

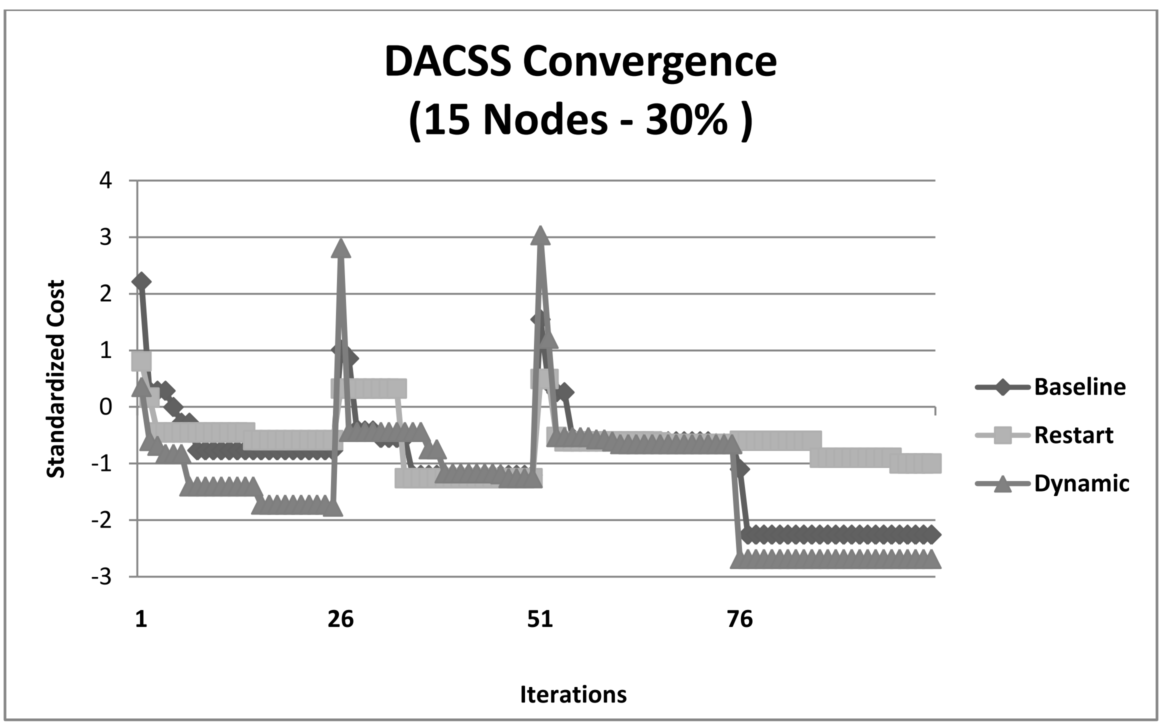


**Figure 63. DACSS Average Convergence (15 Nodes - 30% Change)**

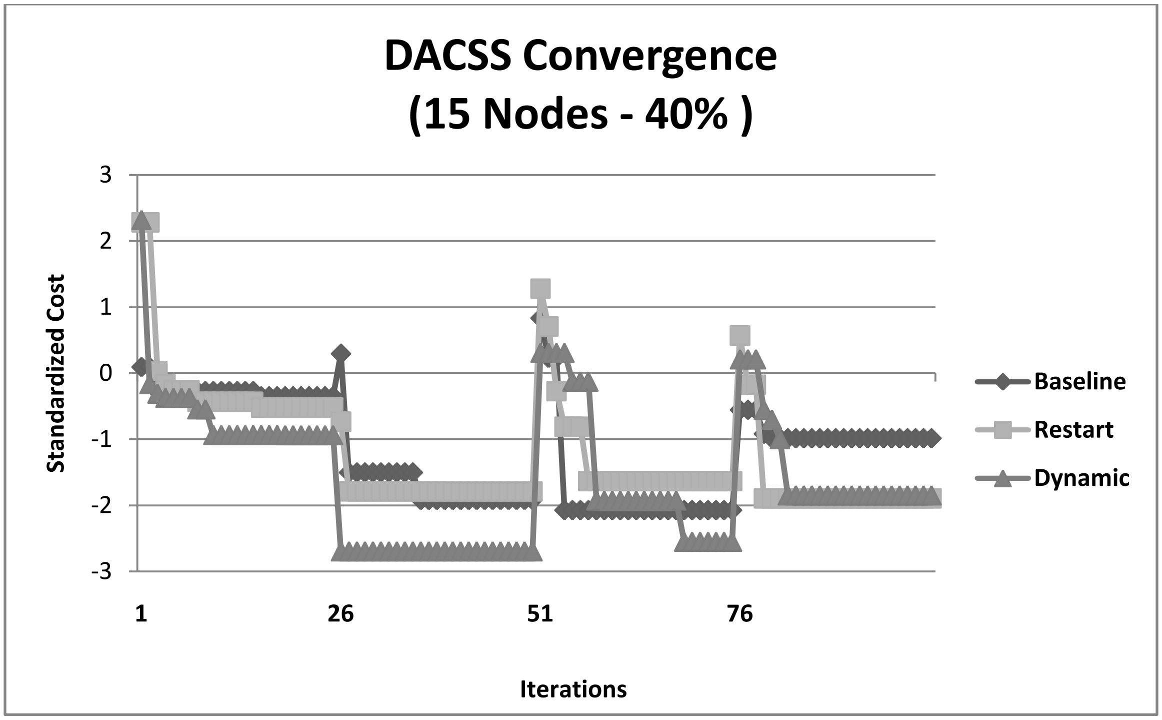


**Figure 64. DACSS Average Convergence (15 Nodes - 40% Change)**

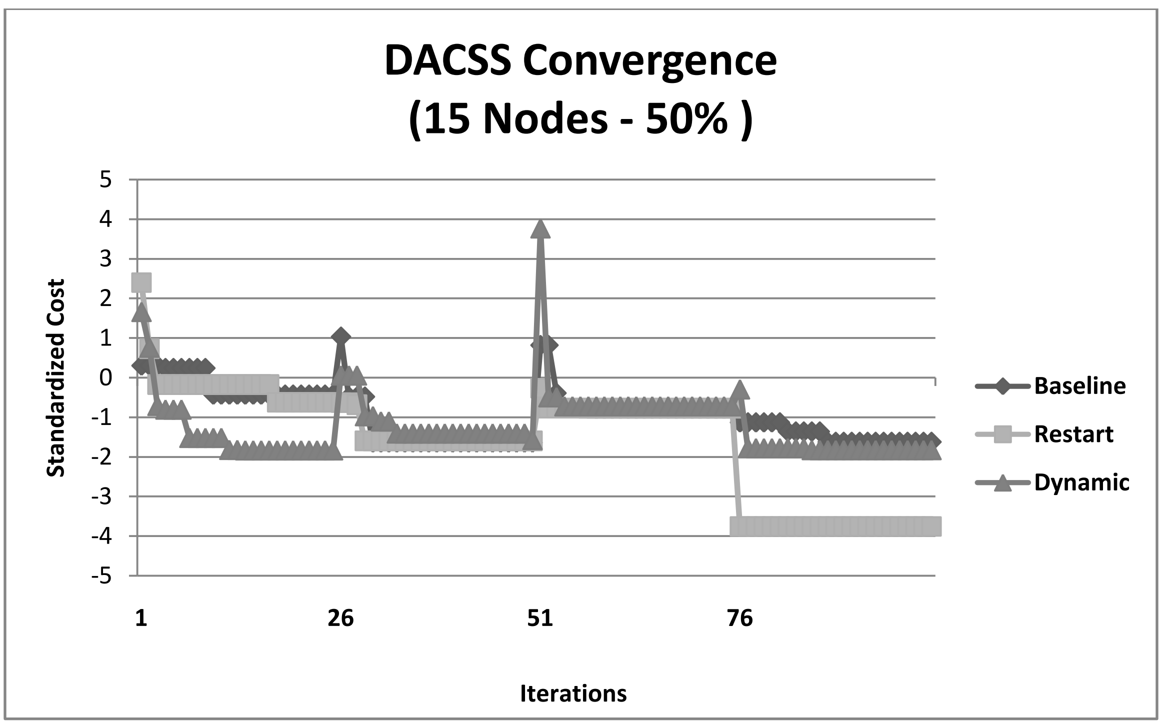


**Figure 65. DACSS Average Convergence (15 Nodes - 50% Change)**

## Bibliography

Aarts, E; Lenstra J. (1997). *Local Search in Combinatorial Optimization.* John Wiley & Sons Ltd.

Ahuja, R., Magnanti, T., & Orlin, J. (1993). *Network Flows: Theory, Algorithms, and Applications.* Upper Saddle River, NJ: Prentice Hall.

Alvarez, A., Gonzalez-Velarde, J., & De-Alba, K. (2005). Grasp Embedded Scatter Search for the Multicommodity Capacitated Network Design Problem. *Journal of Heuristics , 11* (3), 233-257.

Bachem, A., Hochstattle, W., & Malich, M. (1993). Simulated Trading: A New Parallel Approach for Solving Vehicle Problems. *ParCo '93 Parallel Computing Conference.* Grenoble, France.

Beasely, J., & Chu, P. (1996). A Genetic Algorithm for the Set Covering Problem. *European J. Oper. Res.* (94), 392-404.

Boudali, I., W., F., & Ghedira, K. (2004). How to Deal with the VRPTW by using Multi-agent Coalitions. *HIS '04 Hybrid Intelligent System Conference.* Japan.

Boudali, L., Fki, W., & Ghedira, K. (2005). An Interactive Distributed Approach for the Vehicle Routing Problem with Time Windows. *International Journal of Simulation , 6* (10 and 11), 48-59.

Bullnheimer, B., Hartl, R., & Strauss, C. (1999). A New Rank-based Version of the Ant System: A Computational Study. *Central European Journal for Operations Research and Economics , 7* (1), 25-38.

Buntine, W. (1991). Theory Refinement of Bayesian Networks. *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence* (pp. 52-60). Morgan Kaufmann.

Burckert, H., Fischer, K., & Vierke, G. (1998). Transportation Scheduling with Holonic MAS - The TeleTruck Approach. *PAAM '98 Practical Applications of Intelligent Agents and Multi-agents Conf.* London, UK.

Chhikara, R., & Folks, J. (1989). *The Inverse Gaussian Distribution.* Marcel Dekker, Inc.

Chickering, D., Geiger, D., & Heckerman, D. (1996). Learning Bayesian Network is NP-Complete. In D. Fisher, & H. Lenz, *Learning from Data: Artificial Intelligence and Statistics V* (pp. 121-130). Springer-Verlag.

Choudhury, R., Bandyopadhyay, S., & Paul, K. (2000, Aug 11). A Distributed Mechanism for Topology Discovery in Ad Hoc Wireless Networks Using Mobile Agents. *Mobile and Ad Hoc Networking and Computing* , 145-146.

Cooper, G., & Herkovits, E. (1992). A Bayesian Method for the Induction of Probabilistic Networks from Data. *Machine Learning , 9* (4), 309 - 348.

deCampos, L., & Peurta, J. (2001). Stochastic Local and Distributed Search Algorithms for Learning Belief Networks. *III Int Symosium on Adaptive Systems (ISAS): Evolutionary Computation and Probabilistic Graphical Model*, (pp. 109-115).

deCampos, L., Fernandez-Luna, J., Gamez, J., & Puerta, J. (2002, November). Ant Colony Optimizatino for Learning Bayesian Networks. *International Journal of Approximate Reasoning* , 291 - 311.

deCampos, L., Gamez, J., & Puerta, J. (2002). Learning Bayesian Networks by Ant Colony Optimization: Searching in two Different Spaces. *Mathware and Soft Computing , 9* (2-3), 251-268.

Dennis, J. (1964). Distributed Solution of Network Programming Problems. *IEEE Transactions on Communications , 12* (2), 176-184.

Dorigo, M., & Gambardella, L. (1997). Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation , 1*, 53 - 66.

Dorigo, M., & Stutzle, T. (2004). *Ant Colony Optimization.* Cambridge, MA: The MIT Press.

Dorigo, M., Maniezzo, V., & Colorni, A. (1991). *An Autocatalytic Optimization Process.* Milano, Italy: Dipartimento di Elettronica, Politecnico di Milano.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions of Systems, Man and Cybernetics-Part B , 26* (1), 1-13.

Erwin, M. (2006, March). Combining Quality of Service and Topology Control in Directional Hybrid Wireless Networks. *Masters Thesis, AFIT/GOR/ENS/06-07* . Wright Patterson AFB, OH: Air Force Institute of Technology.

Eyckelhof, C., & Snoek, M. (2002). Ant Systems for a Dynamic TSP. *Proceedings of the Third International Workshop on Ant Algorithms. 2463*, pp. 88-99. London: Springer-Verlag.

Fisher, K., & Kuhn, N. (1993). A DAI Approach to Modeling the Transportation Domain. *DFKI-Research Report RR-93-25* .

Fisher, M. (1994). Optimal Solution of Vehicle Routing Problems using Mimimum K-Trees. *Operations Research , 42*, 626-642.

Fisher, T., & Merz, P. (2005). A Distributed Chained Lin-Kernighan Algorithm for TSP Problems. *19th IEEE Internation Parallel and Distributed Processing Symposium*, (p. 16b).

Ford, L., & Fulkerson, D. (1962). *Flows in Networks.* Princeton, NJ: Princeton University Press.

Gardiner, V. (2006). The Air Force's Newest Weapon Systems. Maxwell AFB, AL: Air War College.

Garner, R. (2007). Heuristically Driven Search Methods for Topology Control in Directional Wireless Hybrid Networks. *Masters Thesis, AFIT/GCS/ENG/07-M* . Wright Patterson AFB, OH: Air Force Institute of Technology.

Gendron, B., Crainic, T., & Frangioni, A. (1998). Multicommodity Capacitated Network Design. Norwell, MA: Kluwer.

Guntsch, M., Middendorf, M., & Schmeck, H. (2001). An Ant Colony Optimization Aproach Dynamic TSP. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)* (pp. 860-867). Morgan-Kaufmann Publishers.

Hochbaum, D. (1997). *Approximation Algorithms for NP-Hard Problems.* Boston, MA: PWS Publishing Company.

Jiang, Y., & Jiang, J. (2005, August). A Multi-Agent Coordination for the Variation of Underlying Network Topology. *Expert Systems with Applications , 29* (2), pp. 372-382.

Joshi, A., Mishra, N., Batta, R., & Nagi, R. (2004). Ad Hoc Sensor Network Topology Design for Distributed Fusion: A Mathematical Programming Approach. *7th International Conference on Information Fusion.* Stockholm, Sweden.

Kefi, M., & Ghedira, K. (2004). A Multi-agent Model for the VRPTW. *Urban Transport '04.* Germany.

Kilby, P., Prosser, P., & Shaw, P. (1998). *Dynamic VRPs: A Study of Scenarios.* Technical Report APES-06-1998, University of Stratchclyde.

Kindervater, G., & Savelsbergh, M. (1997). Vehicle Routing: Handling Edge Exchanges. In E. Aarts, & J. Lenstra, *Local Search in Comb inatorial Optimization* (pp. 311-336). Chichester, UK: John Wiley & Sons.

Kleeman, M., Lamont, G. H., & Graham, S. (2007). Solving Multicommodity Capacitated Network Design Problems using a Multiobjective Evolutionary Algorithm. *IEEE Sympsium on Computational Intelligence in Security and Defense Applications*, (pp. 33-41).

Kleinberg, J., & Tardos, E. (2006). *Algorithm Design.* Boston, MA: Addison-Wesley.

Laxmi, V., Jain, L., & Gaur, M. (2006). Ant Colony Optimization Based Routing on NS-2. *International Conference on Wireless Communication and Sensor Networks (WCSN).* India.

Lenstra, J., & K., R. (1981). Complexity of Vehicle Routing and Scheduling Problems. *Networks* , *11*, 221-227.

Llorca, J. e. (2005). Optimizing Performance of Hybrid FSO/RF Networks in Realistic Dynamic Scenarios. *Proceedings of the SPIE Optics and Photonics.*

Lo, C.-C., & Chang, W.-H. (2000). A Multiobjective Hybrid Genetic Algorithm for the Capacitated Multipoint Network Design Problem. *IEEE Transactions on Systems, Man, and Cybernetics* , 461-470.

Lopez, T. S. (2006, December 5). *Air Force Leader to Discuss New 'Cyber Command'.* Retrieved April 10, 2008, from Air Force Print News: http://www.af.mil/news/story.asp?storyID=123028524

Machado, P., Tavares, J., Pereira, F., & Costa, E. (2002). Vehicle Routing Problem: Doing it the Evolutionary Way. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, (p. 690).

Michalewics, Z., & Fogel, D. (2004). *How to Solve It; Modern Heuristics* (2nd ed.). New York: Springer-Verlag.

Montemanni, R., Gambardelaa, L., Rizzoli, A., & Donati, A. (2003). A New Algorithm for a Dynamic Vehicle Routing Problem based on Ant Colony System. *Second Internation Workshop on Freight Transportation and Logistics.* Palermo, Italy.

Murata, T., & Itai, R. (2005). Multi-Objective Vehicle Routing Problems Using Two-Fold EMO ALgorithms to Enhance Solution Similarity on Non-dominated Solutions. In *Evolutionary Multi-Criterion Optimization* (Vol. 3410). Heidelberg: Springer Berlin.

Oimoen, S., Peterson, G., & Hopkinson, K. (2008). Network Formation using Ant Colony Optimization. In M. Dorigo, M. Birattari, C. Blum, M. Clerc, T. Stutzle, & A. Winfield, *Ant Colony Optimization and Swarm Intelligence* (pp. 405-406). Berlin, Germany: Springer.

Osman, I., & Kelly, J. (1996). *Meta-Heuristics: Theory & Applications.* Boston: Kluwer Academic Publishers.

Rajaraman, R. (2002). Topology Control and Routing in Ad hoc Networks: A Survey. *SIGACT News* .

Sakuma, J., & Kobayashi, S. (2007). A Genetic Algorithm for Privacy Preserving Combinatorial Optimization. *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation* (pp. 1372-1379). London, England: ACM.

Stutzle, T., & Hoos, H. (1997). The MaxMin Ant System and the Local Search for the Traveling Salesman Problem. *IEEE International Conference on Evolutionary Computing* (pp. 309-315). Piscataway, NJ: McGraw-Hill.

Thomas, T. J. (2004). Fire Ant: an Algorithm Providing High-Performance Distributed Fault-Tolerant Communication for Internet-Scale Network Survivability. *Doctoral Thesis. UMI Order Number: AAI3160685* . Florida State University.

Toth, P., & Vigo, D. (2001). An Overview of Vehicle Routing Problems. In *The Vehicle Routing Problem* (pp. 1-26). Philadelphia, USA: SIAM, Society for Industrial and Applied Mathematics.

Wenning, B., Timm-Giel, A., & Pesch, D. (2006). A Distributed Routing Approach for Vehicle Routing in Logistic Networks. *IEEE 64th Vehicular Technology Conference,* (pp. 1-5).

Zaleta, N., & Alvarez-Socarras, A. (2004). Tabu Search-based Algorithm for Capacitated Multicommodity Network Design Problem. *14th International Conference on Electronics, Communications, and Computers*, (p. 144).

Zhou, A., Kang, L., & Yan, Z. (2003). Solving Dynamic TSP with Evolutionary Approach in Real Time. *Evolutionary Computation , 2*, 951-957.

| REPORT DOCUMENTATION PAGE | *Form Approved* *OMB No. 074-0188* |
|---|---|



| **1. REPORT DATE** *(DD-MM-YYYY)* | **2. REPORT TYPE** | **3. DATES COVERED** *(From – To)* |
|---|---|---|
| 26-03-2009 | **Ph. D. Dissertation** | Aug 2005 – Mar 2009 |

| **4. TITLE AND SUBTITLE** | |
|---|---|
| DYNAMIC NETWORK FORMATION USING ANT COLONY OPTIMIZATION | **5a. CONTRACT NUMBER** |
| | **5b. GRANT NUMBER** |
| | **5c. PROGRAM ELEMENT NUMBER** |
| **6. AUTHOR(S)** | **5d. PROJECT NUMBER** 08-175 |
| Oimoen, Steven C., Major | **5e. TASK NUMBER** |
| | **5f. WORK UNIT NUMBER** |

| **7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)** | **8. PERFORMING ORGANIZATION REPORT NUMBER** |
|---|---|
| Air Force Institute of Technology<br>Graduate School of Engineering and Management (AFIT/EN)<br>2950 Hobson Street, Building 642<br>WPAFB OH 45433-7765 | AFIT/DCS/ENG/09-06 |
| **9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)** | **10. SPONSOR/MONITOR'S ACRONYM(S)** AFOSR |
| Air Force Office of Scientific Research (AFOSR)<br>875 N. Randolph St.<br>Suite 325, Room 3112<br>POC: Dr. Robert Bonneau<br>Arlington, VA 22203-1768<br>COM: 703-696-9545 Email: robert.bonneau@afosr.af.mil | **11. SPONSOR/MONITOR'S REPORT NUMBER(S)** |

**12. DISTRIBUTION/AVAILABILITY STATEMENT**

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

**13. SUPPLEMENTARY NOTES**

**14. ABSTRACT**

This research presents three contributions for solving highly dynamic (i.e. drastic change within the network) Multi-commodity Capacitated Network Design Problems (MCNDPs) resulting in a distributed multi-agent network design algorithm. The first contribution incorporates an Ant Colony Optimization (ACO) algorithm Ant Colony System (ACS) to solve the static MCNDP with weak constraints. Second, a new algorithm is developed and has the capability to dynamically adjust its exploration parameter of the solution space. This enhanced algorithm converges quickly and automatically adjusts to the dynamically changing network environment. Third, a distributed approach is created replacing the previous centralized solver. The distributed algorithm produces comparable results, but more importantly calculates the network topology in less than 20 percent of the computation time.

**15. SUBJECT TERMS**

Ant Colony Optimization (ACO), Dynamic Multi-commodity Capacitated Network Design Problem (DMCNDP), distributed processing

| **16. SECURITY CLASSIFICATION OF:** | | | **17. LIMITATION OF ABSTRACT** | **18. NUMBER OF PAGES** | **19a. NAME OF RESPONSIBLE PERSON** |
|---|---|---|---|---|---|
| **a. REPORT** | **b. ABSTRACT** | **c. THIS PAGE** | | | Dr. Gilbert L. Peterson (ENG) |
| **U** | **U** | **U** | **UU** | 315 | **19b. TELEPHONE NUMBER** *(Include area code)*<br>(937) 255-6565, ext 4281<br>Email: gilbert.peterson@afit.edu |